Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Luka Krivec

Ogrodje za izbiro pristopov za izboljšavo procesa razvoja informacijskih sistemov

magistrsko delo

magistrski program druge stopnje
računalništvo in informatika

Mentor: doc. dr. Damjan Vavpotič

Ljubljana, 2016

Izjava o avtorstvu magistrskega dela

Spodaj podpisani Luka Krivec sem avtor magistrskega dela z naslovom:

Ogrodje za izbiro pristopov za izboljšavo procesa razvoja informacijskih sistemov

S svojim podpisom zagotavljam, da:

- sem magistrsko delo izdelal samostojno pod mentorstvom doc. dr. Damjan Vavpotič
- so elektronska oblika magistrskega dela, naslov (slovenski, angleški), povzetek (slovenski, angleški) ter ključne besede (slovenske, angleške) identični s tiskano obliko magistrskega dela,
- soglašam z javno objavo elektronske oblike magistrskega dela v zbirki "Dela FRI".

V Ljubljani, 28. decembra 2016 Podpis avtorja:

Zahvala

Zahvaljujem se svoji družini, ki mi je vsa leta študija stala ob strani. Hvala tudi sošolcem in prijateljem, ki so mi pomagali pri študiju. Za pomoč in nasvete pri pisanju magistrskega dela se zahvaljujem mentorju doc. dr. Damjanu Vavpotiču.

Luka Krivec, 2016

Kazalo

Seznam uporabljenih kratic

kratica	angleško	slovensko
IS	Information Systems	Informacijski sistemi
ME	Method Engineering	Konstruiranje metodologij
SME	Situational Method Engineering	Situacijsko konstruiranje metodologij
ISDM	Information systems development methodology	Metodologija razvoja informacijskih sistemov
GUI	Graphical user interface	grafični uporabniški vmesnik
UML	Unified modeling language	poenoteni jezik modeliranja
WMS	Warehouse Management System	sistem za upravljanje skladišč
SQL	Structured query language	strukturiran povpraševalni jezik za delo s podatkovnimi bazami
VPN	Virtual private network	navidezno zasebno omrežje

Povzetek

V magistrskem delu smo razvili ogrodje za izbiro pristopov za izboljšavo procesa razvoja informacijskih sistemov. Ogrodje združuje področji evalvacijskih modelov in konstruiranja metodologij. Podan je pregled nad obstoječimi evalvacijskimi modeli in različnimi pristopi h konstruiranju metodologij. V ogrodje smo vključili evalvacijski model, ki omogoča evalvacijo procesa razvoja informacijskih sistemov s sociološkega, tehničnega in ekonomskega vidika. Z evalvacijskim delom našega ogrodja najprej evalviramo elemente procesa razvoja informacijskih sistemov ter tako pridobimo seznam elementov, ki jih je potrebno izboljšati. Sledi izbira pristopov za izboljševanje elementov razvojnega procesa, ki temelji na uveljavljenih tehnikah s področja konstruiranja metodologij. Za elemente, ki jih je treba izboljšati, najprej poiščemo več smiselnih alternativ elementov, ki praviloma izhajajo iz uveljavljenih razvojnih procesov. Z uporabo izbirnega dela ogrodja ocenimo primernost alternativnih elementov za projekt in razvojno skupino ter na tej osnovi pripravimo seznam primernih alternativnih elementov. Na koncu najboljše alternative predstavimo vodstvu podjetja, ki izbere alternative za vpeljavo v razvojni proces podjetja. Delovanje ogrodja smo preverili v okviru študije primera, ki smo jo izvedli v slovenskem podjetju, ki se ukvarja z informatiko v logistiki. Vodstvo podjetja je potrdilo, da so predstavljeni rezultati za podjetje uporabni.

Ključne besede

informacijski sistemi, konstruiranje metodologij, razvoj programske opreme, evalvacijski modeli

Abstract

In this master's thesis, we developed a framework for selection of approaches to improve the information systems development process. The framework combines both evaluation models and method engineering. It presents an overview of currently developed evaluation models and different approaches to method engineering. Our framework includes an evaluation model that allows the evaluation of the information system development process from social, technical and economic perspectives. With evaluation part of our framework, we first evaluate elements of information system development process, and so obtain a list of elements that should be improved. This is followed by the selection of approaches for improvement of information systems development process, based on established techniques from the field of method engineering. First, we find meaningful alternatives for elements that need improvement, and these are usually based on established development processes. Using the selection part of our framework we assess the appropriateness of alternative elements for the project and development team, and on this basis prepare a list of suitable alternative elements. We finally present the best alternatives to the company management, which then chooses those that will be introduced. We tested our framework with a case study carried out in a Slovenian company, which deals with informatics in logistics. The managers confirmed that the results of this study were useful for their company.

Keywords

information systems, method engineering, software engineering, evaluation models

Poglavje 1

Uvod

Informacijski sistemi se uporabljajo na številnih področjih in se morajo stalno prilagajati novim zahtevam okolja. Skozi leta je bila razvita vrsta metodologij za razvoj informacijskih sistemov, ki omogočajo optimizacijo procesov razvoja informacijskih sistemov. Metodologije predpisujejo splošen način, kako načrtovati in izvajati razvoj informacijskih sistemov. Že avtorji metodologij so spoznali, da ne obstaja en sam pristop, ki bi bil primeren za vse tipe projektov in organizacij. Hkrati pa so z napredovanjem tehnologij nastajale potrebe po novih metodoloških pristopih. Zato se je začelo razvijati raziskovalno področje konstruiranja metodologij (angl. method engineering). Področje se ukvarja z načrtovanjem, izgradnjo in prilagajanjem metodologij organizacijam in projektom, tehnikami in orodji za razvoj sistemov. Ne osredotočajo se na splošne metodologije, ampak na izgradnjo organizacijsko ali projektno specifične metodologije. Do sedaj so bile razvite številne metodologije izvajanja procesa razvoja informacijskih sistemov. Metodologije se razlikujejo v formalnosti, kompleksnosti, številu in načinu izvajanja opravil ter številnih drugih parametrih. Tipično se organizacije znajdejo pred problemom, da nobena od splošnih metodologij v popolnosti ne ustreza njihovim potrebam. Številne organizacije se odločijo, da bodo uporabljale tako imenovan *ad-hoc* pristop. Pri tem pristopu se postopek razvoja razlikuje od primera do primera. Eden od problemov, ki se pri tem pojavijo, je, da se običajno postopki ne beležijo in se zaradi tega proces ne more izboljševati na podlagi preteklih dognanj. Na drugi strani imamo več komercialnih metodologij, ki se še vedno uporabljajo, vendar so bile spoznane kot nefleksibilne. Razvojne ekipe jih zato velikokrat ocenijo kot neprimerne in jih posledično nočejo uporabljati. Konstruiranje metodologij pri tem problemu pomaga z izgradnjo metodologij, ki so prilagojene projektom ali organizaciji. Pri tem gre lahko za izgradnjo nove metodologije ali združevanje pristopov različnih metodologij.

Prvi korak pri postopku izboljšave procesa razvoja informacijskih sistemov je evalvacija trenutnega procesa razvoja informacijskih sistemov. Z evalvacijo želimo pridobiti

ocene za različne aktivnosti, opravila, dokumente in orodja, ki se trenutno uporabljajo. Za aktivnosti, opravila, dokumente in orodja, v nadaljevanju uporabljamo izraz elementi procesa razvoja informacijskih sistemov. Želimo izvedeti, kako trenutno poteka izvajanje elementov procesa razvoja informacijskih sistemov in kako so s trenutnim izvajanjem zadovoljni uporabniki procesa. Evalvacija mora biti čim bolj popolna in se prilagajati različnim sodelujočim v procesu ter njihovim specifičnim pogledom na proces. Za ocenjevanje kakovosti procesa razvoja informacijskih sistemov obstajajo številni evalvacijski modeli, ki nam pomagajo pridobiti oceno kakovosti razvojnih procesov. Modeli nam omogočajo evalvacijo z vidika različnih deležnikov na nivoju skupine aktivnosti ali na nivoju posameznih aktivnosti. Obstajajo številne raziskave na področju evalvacije tehničnega in sociološkega vidika elementov procesa razvoja informacijskih sistemov. Problem, ki smo ga identificirali, je, da nekateri evalvacijski modeli vidike obravnavajo nepovezano. Zato smo se v okviru magistrske naloge osredotočili na model [1, 2], ki omogoča povezano obravnavo tehničnega, sociološkega in ekonomskega vidika za elemente procesa razvoja informacijskih sistemov.

V magistrski nalogi smo razvili ogrodje, ki združuje evalvacijski model in tehnike iz konstruiranja metodologij. S tem smo pridobili nov, bolj celovit pristop k izboljševanju procesa razvoja informacijskih sistemov. Ogrodje podpira evalvacijo trenutnega procesa razvoja informacijskih sistemov sočasno s sociološkega, tehničnega in ekonomskega vidika ter omogoča pripravo projektno in organizacijsko specifičnih pristopov za izboljšavo le tega. Ključni doprinos naloge je torej ogrodje, ki na podlagi rezultatov evalvacije trenutnega razvojnega procesa ter karakteristik projekta, z uporabo principov konstruiranja metodologij, pripravi seznam priporočenih izboljšav posameznih elementov procesa razvoja informacijskih sistemov.

V prvem poglavju dela je podan pregled literature in ključnih področij. Podrobneje so predstavljeni evalvacijski modeli in področje konstruiranja metodologij. Razložen je pomen evalvacijskih modelov za potrebe izboljševanja procesov razvoja informacijskih sistemov in podan opis nekaterih ključnih evalvacijskih modelov na tem področju. Opisani in razloženi so posamezni zrelostni nivoji modela za ocenjevanje zrelosti razvojnega procesa CMMI (angl. Capability Maturity Model Integration). Nato je opisan evalvacijski model, ki je bil predstavljen v delih [1, 2] in je uporabljen v našem ogrodju. Opisan je postopek uporabe evalvacijskega modela in karakteristike, po katerih poteka ocenjevanje elementov. Prikazano je tudi, kako izgleda grafična predstavitev rezultatov tega evalvacijskega modela in razloženo, kako poteka klasifikacija elementov v skupine. V drugem poglavju sledi opis področja konstruiranja metodologij. Zaradi različnih izrazov, ki se uporabljajo na področju, so najprej predstavljeni ključni pojmi, nato pa sledi predstavitev izbranih tehnik z omenjenega področja. Razloženih je več načinov za načrtovanje metod razvoja informacijskih sistemov. Opisane so različne tehnike, ki omogočajo pripravo projektno oziroma organizacijsko specifičnih metodologij. V tretjem poglavju je podrobno opisano

naše ogrodje. Na začetku je prikazana zgradba celotnega ogrodja, nato pa so podrobneje opisani njegovi deli: pripravljalni del, evalvacijski del, izbirni del in potrditveni del. Pri vsakem delu ogrodja je razloženo, kateri postopki se uporabljajo in kaj so rezultati posameznega dela. Pri evalvacijskem delu je predstavljen vprašalnik in opisan postopek razvrščanja elementov v skupine. Predstavljene so tudi projektne karakteristike in postopek izbire elementov v izbirnem delu ogrodja. V zaključku poglavja predstavljamo še tipičen postopek uporabe ogrodja. V četrtem poglavju smo predstavili študijo primera. Ogrodje je bilo testirano na primeru podjetja. Poglavje vsebuje opis podjetja in predstavitev obstoječega procesa razvoja informacijskih sistemov. Faze procesa razvoja so podrobno opisane in grafično predstavljene. Na koncu poglavja so celovito predstavljeni rezultati študije primera ter na konkretnih primerih predstavljen postopek uporabe našega ogrodja. V okviru potrditvenega dela ogrodja, so bili rezultati študije primera preverjeni skupaj z vodstvom podjetja. V zaključku povzamemo ključne prispevke magistrskega dela, predstavimo njegove omejitve in možne nadaljnje izboljšave ogrodja v prihodnosti.

Poglavje 2

Pregled literature in ključnih področij

V nadaljevanju je podan pregled literature in ključnih področij, ki so povezani s tematiko magistrskega dela. Predstavljeni sta področji evalvacijskih modelov in konstruiranja metodologij. Iz vsakega področja je predstavljenih nekaj pomembnejših obstoječih pristopov, ki so nudili osnovo tudi za oblikovanje našega ogrodja.

2.1 Evalvacijski modeli

Evalvacijski model nam pomaga razumeti korake, ki so potrebni za kakovostno evalvacijo. Pri evalvaciji metodologije razvoja informacijskih sistemov (ISDM) je bilo razvitih veliko evalvacijskih modelov, ki se osredotočajo na različne vidike procesa. Eden izmed modelov za ocenjevanje zrelosti razvojnega procesa je CMMI (angl. Capability Maturity Model Integration) [3]. Model opisuje različne razvojne stopnje procesa od začetne stopnje do optimizirajoče stopnje. Model je sestavljen iz petih nivojev zrelosti:

- Začetni nivo: organizacija ne ponuja stabilnega okolja za razvijanje in vzdrževanje programske opreme. Proces razvoja programske opreme organizacije začetnega nivoja je nepredvidljiv, ker se razvojni proces neprestano spreminja s potekom dela.
- Ponovljiv nivo: pri ponovljivem nivoju so vzpostavljeni osnovni procesi upravljanja projektov, ki omogočajo spremljanje stroškov, trajanja razvoja in učinkovitosti dela. Uspešnost projektov je v veliki meri odvisna od rezultatov preteklih projektov.
- Definiran nivo: pri definiranem procesu je dokumentiran in standariziran skupen proces razvoja organizacije. Ta poleg upravljanja vključuje tudi vse tehnične ak-

tivnosti. Pri razvoju projektov se uporablja prilagojen standardni proces razvoja programske opreme, katerega ustreznost je organizacija pred tem potrdila.

- Upravljan nivo: pri tem nivoju imajo organizacije vzpostavljene kvantitativne kvalitetne cilje za razvojne produkte in procese. Merjena je produktivnost in kvaliteta za izboljšanje aktivnosti razvojnega procesa preko vseh projektov organizacije. Identificirani so tudi podprocesi, ki pripomorejo v večji splošni učinkovitosti procesa. Razvojni proces je predvidljiv, ker je merjen in upravljan v okviru meritvenih limitov.

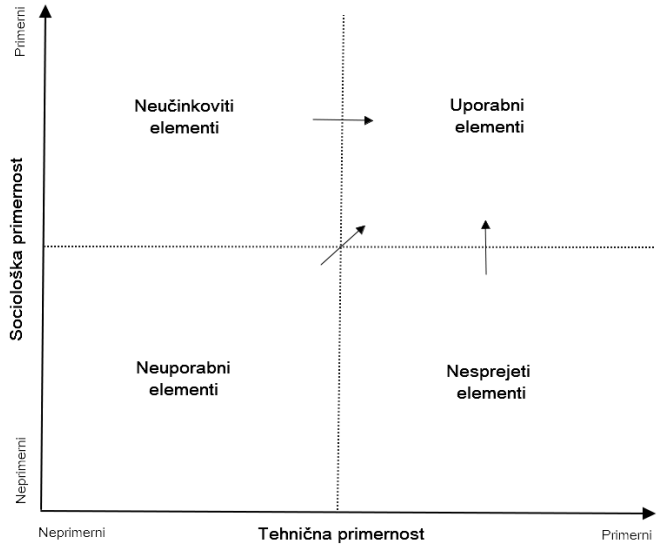
- Optimizirajoč nivo: pri optimizirajočem nivoju je celotna organizacija fokusirana na konstantno izboljševanje razvojnega procesa. Organizacija odkriva slabosti procesa in izboljšuje proces proaktivno. Zrelost razvojnih procesov v takih organizacijah se nenehno izboljšuje.

Ogrodje, ki smo ga razvili v okviru magistrske naloge, bo pomagalo organizacijam pri doseganju višje stopnje zrelosti po modelu CMMI.

V magistrski nalogi izhajamo iz modela, predstavljenega v delih [1, 2]. Ta model je zgrajen na osnovi obstoječih evalvacijskih modelov, uporabljenih na področju metodologije razvoja informacijskih sistemov (ISDM). Medtem ko se veliko evalvacijskih modelov osredotoča samo na en vidik ISDM, pa ta model združuje več vidikov. To so tehnični, sociološki in ekonomski vidik. Pomembnost pogleda z vseh vidikov na vsak element v ISDM je, da ugotovimo pomanjkljivosti elementa za posamezen vidik. Raziskave [4, 5, 6] so pokazale, da je dovolj velik razlog za zavrnitev elementa ISDM neustreznost v enem od vidikov, kljub temu da je v drugih vidikih povsem ustrezen.

Organizacije, ki razvijajo programsko opremo, običajno upoštevajo določena pravila in postopke pri razvoju. Nekatere organizacije uporabljajo formalne ISDM, druge pa se zanašajo na neformalne dogovore med razvijalci. Tudi tiste organizacije, ki uporabljajo formalne ISDM, jih redko uporabljajo striktno [7]. Raziskovalci so odkrili, da sta zato pomembna dva vidika. Tehnični vidik se nanaša na neprilagojenost formalnih ISDM na specifične organizacijske in projektne potrebe. Sociološki vidik izpostavlja neujemanje sociooloških karakteristik razvojne ekipe in organizacije formalnim ISDM. Poleg teh dveh vidikov je treba upoštevati še ekonomski vidik, ki je ključnega pomena za upravljalske odločitve [2].

Model [1, 2] ne evalvira procesa kot celote, ampak kot kompozicijo med seboj povezanih elementov. Upošteva samo elemente, ki so lahko formalizirani. Primer takih elementov so aktivnosti, vloge, artefakti, tehnike, predloge, navodila itd. Po evalvaciji so elementi pozicionirani v razsevnem diagramu. Diagram je razdeljen na štiri kvadrante. Prvi kvadrant predstavlja neučinkovite in nesprejete elemente, drugi kvadrant neučinkovite, ampak



Slika 2.1: Kvadranti razsevnega diagrama evalvacijskega modela [1, 2]

sprejete elemente, tretji kvadrat nesprejete, ampak učinkovite elemente, in četrti kvadrant sprejete in učinkovite elemente, kot predstavlja slika 2.1.

Elementi so v diagram razvrščeni na podlagi odgovorov na vprašanja o frekvenci uporabe in frekvenci priložnosti uporabe, ki so evalvirana s sedemtočkovno Likertovo lestvico. Diagram je na kvadrante razdeljen na podlagi povprečne ocene odgovor na vprašanja. Ekonomski vidik je prikazan kot številska ocena ob posameznem elementu.

Kategorizacija elementov v kvadrante razsevnega diagrama olajšuje pripravo izboljševalnega scenarija. Izboljševalni scenarij je definiran kot seznam konkretnih akcij, ki morajo biti izvedene, da se izboljša stopnja učinkovitosti in sprejetosti elementa ISDM. Pri pripravi izboljševalnega scenarija moramo za elemente v prvem kvadrantu preučiti smiselnost uporabe, pri elementih v drugem kvadrantu se osredotočiti na ustvarjanje več priložnosti za uporabo, za elemente v tretjem kvadrantu poskušati izboljšati sprejetost elementov, elemente v četrtem kvadrantu, ki so že ustrezni, pa samo nadzorovati. Pri zasnovi izboljševalnega scenarija je treba upoštevati, da vseh elementov ne moremo spreminjati v enakem obsegu. Na primer nekatere ključne elemente procesa lahko spreminjamo samo v sociološkem vidiku, da preprečimo nastanek nekonsistentnih metod. Zato model upošteva tudi dimenzijo spremenljivosti pri zasnovi izboljševalnega scenarija. Po izvedbi izboljševalnega scenarija pričakujemo, da se bo večina elementov premaknila v kvadrant z uporabnimi elementi.

Za ocenjevanje elementov ISDM so v modelu [1] uporabili karakteristike, ki so porazdeljene v dve množici. Prva množica so karakteristike, ki merijo trenutno situacijo, druga pa so karakteristike, ki merijo razloge za trenutno stanje elementov.

Karakteristiki **sociološkega vidika**, ki merita trenutno stanje:

- **Frekvenca priložnosti uporabe** meri, kako pogosto razvijalci uporabijo določen element ISDM v primeru, da se pojavi priložnost za uporabo.
- **Konsistentnost uporabe** meri, kako konsistentno razvijalci sledijo navodilom in pravilom določenega elementa razvoja programske opreme.

Karakteristike **sociološkega vidika**, ki merijo razloge za trenutno stanje:

- **Relativna prednost** uporabe elementa meri mnenje uporabnikov, v kolikšni meri element pripomore k boljšemu delu.
- **Sociološka kompatibilnost** meri uporabniška mnenja o ujemanju karakteristik elementov z njihovimi izkušnjami, znanjem in potrebami.
- **Kompleksnost** meri stopnjo težavnosti uporabe elementa.
- **Uporabniške izkušnje in znanje na področju sistemskega razvoja** meri stopnjo znanja in izkušenj uporabnikov na področju programskih orodij, tehnologij, platform, programskih jezikov itd.
- **Prostovoljnost** predstavlja mnenja uporabnikov o tem, ali je element opcijski. Študije ugotavljajo, da v primeru, ko vodstvo ne predpiše uporabe elementa ISDM, ga razvijalci ne uporabljajo.
- **Podpora vodstva** za element meri stopnjo, do katere vodstvo aktivno podpira uporabo elementa.
- **Subjektivna norma** meri stopnjo za katero uporabniki mislijo, da bi morali drugi uporabniki, ki so njim pomembni, uporabiti določen element.
- **Predstavljivost rezultatov** prednosti uporabe elementa meri stopnjo, pri kateri so prednosti uporabe elementa oprijemljive.
- **Vidnost** meri stopnjo, do katere lahko uporabniki opazujejo uporabo drugih uporabnikov določenega elementa.
- **Dostopnost do znanja** meri, kako lahko je pridobiti znanje o določenem elementu.

- **Zadovoljstvo uporabnikov** meri, kako zadovoljni so uporabniki z uporabo določenega elementa [8].

Karakteristike **tehničnega vidika**, ki merijo trenutno stanje:

- **Frekvenca uporabe** meri, kako pogosto nastopi priložnost za uporabo elementa, ne glede na to, ali je element dejansko uporabljen s strani potencialnih uporabnikov.
- **Posledice elementa na sistem, ki bo implementiran**: popolnost, uporabnost, zanesljivost, vzdrževalnost in učinkovitost novega sistema.
- **Posledice elementa na projekt**: poraba časa, strošek projekta, kontrola projekta, kakovost projektnega plana itd.
- **Posledice elementa na uporabnike ISDM**: dvoumnost komunikacije, razumevanje odgovornosti in dolžnosti, omogočanje sodelovanja in podpora treniranja.
- **Posledice elementa na uporabnike**: povečanje zaupanja uporabnika v organizacijo, izboljšanje zadovoljstva.

Karakteristike **tehničnega vidika**, ki merijo razloge za trenutno stanje:

- **Primernost za projekt in sistem** meri stopnjo ujemanja določenega elementa različnim projektnim in sistemskim parametrom, kot npr. kompleksnost, prioriteta, tip.
- **Primernost za razvojno ekipo** meri, kako dobro element ustreza izkušnjam in znanju razvojne ekipe.
- **Primernost za uporabnika** meri stopnjo ustrezanja elementa ISDM vnaprej določenim zahtevam in potrebam uporabnikov.
- **Skladnost z modernimi metodami razvoja** meri, kako aktualen je element glede na tehnologijo, tehnike in priporočila, ki se trenutno uporabljajo na tem področju.
- **Kompatibilnost z informacijsko tehnologijo in programskim okoljem** za sistemski razvoj na določenem projektu meri, kako kompatibilen je določen element ISDM s tehnologijami in programskih okoljem, uporabljenim na določenem projektu.
- **Kompatibilnost z internimi standardi ISDM** meri stopnjo ujemanja elementa z internimi standardi ISDM in splošnimi standardi drugih ISDM.

- **Kompatibilnost s splošnimi standardi** meri stopnjo ujemanja elementa ISDM s splošno definiranimi standardi na tem področju, npr. tehnike modeliranja, standard kodiranja, arhitekturni standardi in vzorci itd.

Karakteristike ekonomskega vidika, ki izhajajo iz modela [2]:

- **Stroški**: merimo, kako element ISDM vpliva na stroške projekta.
- **Cilji**: merimo, kako element ISDM učinkuje na produktivnost podjetja in na zmožnost sledenja ključnim ciljem podjetja.
- **Produkt**: merimo, kako element ISDM pripomore na deležnike z vplivom na izboljšanje produkta in storitev podjetja.

2.2 Konstruiranje metodologij

Konstruiranje metodologij je področje, ki se ukvarja z načrtovanjem, izgradnjo in prilagajanjem metodologij organizacijam in projektom ter tehnikami za razvoj sistemov. Glavno področje konstruiranja metodologij je situacijsko konstruiranje metodologij, ki se osredotoča na kreiranje razvojnih metodologij za specifično situacijo, običajno specifične projektne karakteristike [9]. Konstruiranje metodologij in situacijsko konstruiranje metodologij se ne osredotočata na pripravljene metode ponudnikov, ampak na izgradnjo organizacijsko ali projektno specifičnih metodologij. Izgradnja metodologij je izvedena z izbiro delov metod, ki so bili ustvarjeni in shranjeni v repozitoriju metod.

2.2.1 Terminologija

Na področju konstruiranja metodologij avtorji uporabljajo različno terminologijo, zato so spodaj opisani ključni pojmi.

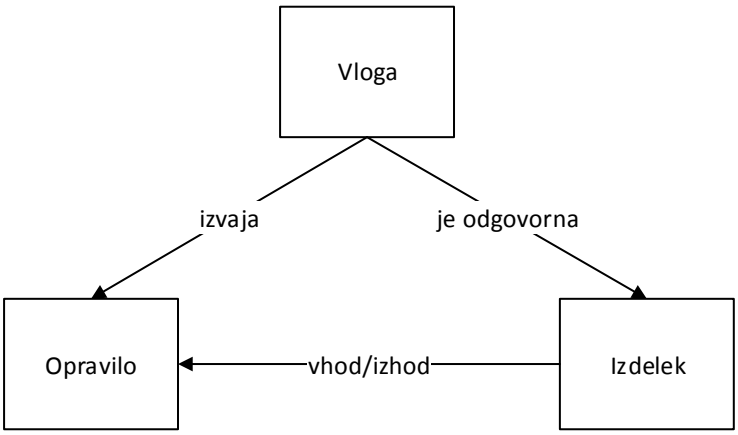
- **Metodologija** je pristop k izvajanju projekta razvoja programske opreme. Zgrajena je na osnovi načina razmišljanja, navodil, pravil in hevristik, ki so sistematično strukturirane v obliki razvojnih aktivnosti s pripadajočimi delovnimi produkti in razvojnimi vlogami [10]. Pogosto se namesto izraza metodologija uporablja tudi izraz metoda. Metodologija se pogosto razume kot združitev več metod.
- **Proces** razvoja informacijskih sistemov je način izvajanja razvoja informacijskih sistemov. Pri izvajanju procesa se morajo razvijalci programske opreme zavedati tudi ljudi in orodij, ki so vključeni. Za to skupno kombinacijo bomo uporabili besedo metodologija.

- **Fragmenti oz. kosi metod**: pri situacijskem konstruiranju metodologij je metodologija zgrajena iz manjših delov. Za to se uporabljata izraza fragment metodologije (angl. method fragment), ki predstavlja element programske opreme. Je osnovni element metodologije, medtem ko je kos metodologije (angl. method chuck) kombinacija enega procesno usmerjenega fragmenta in enega produktno usmerjenega fragmenta.

- **Baza metod**: ko so fragmenti oz. kosi metodologij enkrat pridobljeni, so shranjeni v bazi metod. Iz baze so potem kopirani za kreiranje specifične metodologije na podlagi projektnih karakteristik.

2.2.2 Načrtovanje metod

Načrtovanje metod razvoja informacijskih sistemov vsebuje konceptualno definicijo vseh elementov, ki so del metode. To so aktivnosti, opravila, dokumenti in orodja. Do sedaj so bile metode razvoja programske opreme opisane v pisni obliki, v zadnjem času pa so različni avtorji definirali več jezikov za specifikacijo metod. Prav tako je bilo nekaj poskusov standardizacije. Ti učinki so lahko vidni v standardih ISO/IEC 24744 in SPEM 2.0 [11]. Poleg tega lahko omenimo še področje upravljanja poslovnih procesov (angl. Business Process Management - BPM). Primer standarda na tem področju, ki ga lahko uporabimo tudi za načrtovanje metod, je BPMN 2.0. Do sedaj največkrat uporabljen standard za načrtovanje metod je SPEM 2.0 (angl. Software and System Process Engineering Metamodel) [12]. To je standariziran jezik, ki je bil uradno objavljen s strani OMG (angl. Object Management Group) prvič leta 2002. Šest let pozneje, leta 2008, so izdali drugo verzijo, ki naj bi odpravila probleme prve verzije standarda. Ta verzija je bila široko sprejeta in velikokrat uporabljena. Skupina OMG definira SPEM 2.0 kot procesno inženirski metamodel in konceptualno ogrodje, ki ponuja koncepte za modeliranje, dokumentiranje, predstavljanje, upravljanje in izmenjavo razvojnih metod in procesov. Jedro metamodela SPEM 2.0 temelji na treh osnovnih elementih: opravilo, vloga in izdelek. Povezava med elementi je prikazana na sliki 2.2. Opravila, ki predstavljajo osnovno enoto dela, so izvajana s strani vlog. Vloge metod, ki definirajo množico povezanih spretnosti, kompetenc in odgovornosti, so uporabljene na opravilih, ki definirajo, kdo jih izvaja in izdelke, za katere so vloge odgovorne. Izdelki, ki predstavljajo artifakte, so uporabljeni, izdelani ali modificirani med uporabo metode in so vezani na opravilo za specifikacijo vhodov in izhodov opravil.



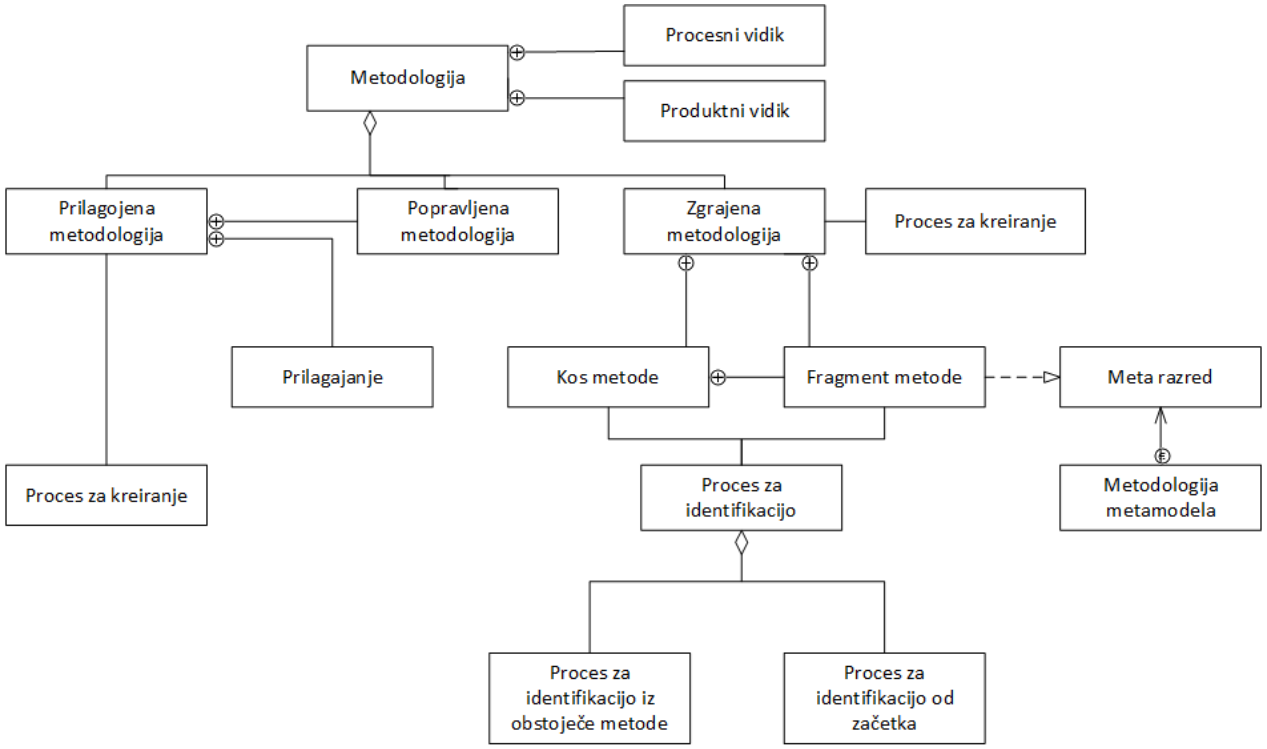
Slika 2.2: Ključni elementi SPEM 2.0

2.2.3 Pregled konstruiranja metodologij

Izzivi konstruiranja metodologij so, kako ustvariti dele metod, kako jih shraniti in pridobiti, kako sestaviti celo metodologijo iz delov metod in kako formalizirati shranjene fragmente. Ker je znanje o metodah fragmentirano in visoko strukturirano, so lahko dodatni fragmenti, ki so prilagojeni za specifično organizacijo ali dodani kot tehnološke spremembe, enostavno dodani v repozitorij metod. Iz praktičnega vidika pa se je pojavil problem pomanjkanja empiričnih podatkov o metodah. Prav tako je bilo ugotovljeno, da je veliko opravil situacijskega kontruiranja metodologij kompleksno [13]. Veliko avtorjev se ukvarja tudi z avtomatizacijo procesov situacijskega konstruiranja metodologij. Harmsen [14] in Odell [15] predlagata, da naj vsako orodje CAME (angl. Computer-Aided Method Engineering) vsebuje naslednje funkcije:

- definicijo in evalvacijo nepredvidljivih pravil in faktorjev,
- shrambo metodoloških fragmentov,
- pridobitev in kompozicijo fragmentov metodologij,
- validacijo in verifikacijo generirane metodologije,
- adaptacijo generirane metodologije,
- integracijo z orodji meta-CASE,
- vmesnik z bazo metod.

Pregled nad konstruiranjem metodologij je strjen v sliki 2.3, kot objektno usmerjen razredni diagram. Znaka plus v krogu in epsilon v krogu sta OML (angl. OPEN Modeling Language [16]) ikoni za konfiguracijo in ne-konfiguracijo, uporabljena, ker UML ne podpira zvez. Znak za konfiguracijo predstavlja, da se določen element uporablja kot konfiguracija drugega elementa, znak za ne-konfiguracijo pa pomeni, da element vsebuje ta element, vendar se ne uporablja za konfiguracijo elementa. Metodologije imajo tako procesni kot produktni vidik. Lahko so zgrajene, prilagojene ali popravljene. V tem delu bodo največkrat omenjene zgrajene metodologije. To so tipično metodologije, ki so sestavljene iz delov metodologij ali fragmentov metodologij, ki so definirani v okviru metamodela. Proces izgradnje vsebuje proces identifikacije. Ta proces lahko poteka od začetka ali izhaja iz obstoječe metodologije.



Slika 2.3: Visokonivojski pregled situacijskega konstruiranja metodologij [17].

2.2.4 Proces kreiranje metodologije iz fragmentov

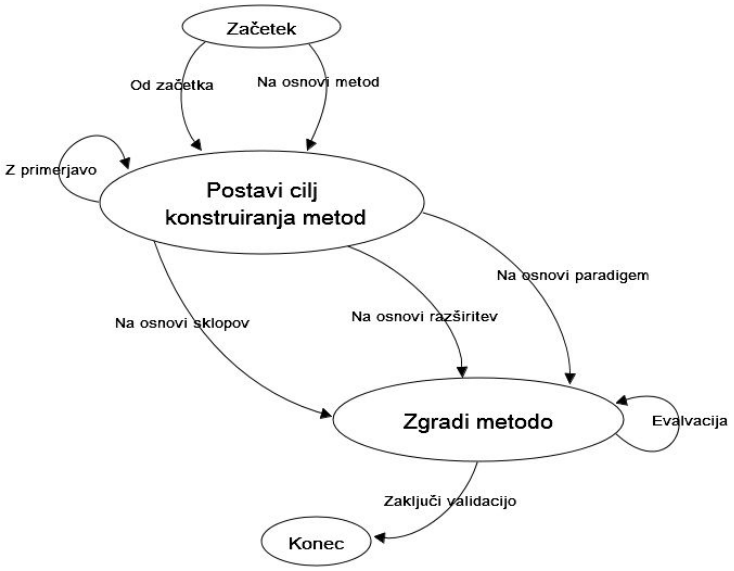
Generični procesni model

Generični procesni model [18] na sliki 2.4 opisuje proces, po katerem je kreiran ISDM kot rezultat situacijskega konstruiranja metodologij. Ta procesni model vsebuje tri vrste pristopov. To so pristopi, ki temeljijo na sklopih, razširitvah ali paradigmah. Pri pristopu, ki temelji na sklopih, se osredotočamo na sprotno grajenje metod, ki se čim bolj prilagajajo projektu. Pristop vsebuje izbiro metod iz repozitorija metod glede na trenutne projektne zahteve in združevanje metod. Pri pristopu, ki temelji na razširitvah, uporabimo vzorce na obstoječih metodah. Pristop, ki temelji na paradigmah, je lahko zgrajen abstraktno, ustvarjen iz metamodela ali prilagojen. Pri tem pristopu ustvarimo produktni in procesni model. S tem omogočimo pogled na metode iz produktnega in procesnega vidika. Te trije pristopi so razširjeni s četrtim, ki je *ad-hoc*, predstavljen v [19].

Model ima dva glavna namena, pomaga določiti cilj konstruiranja metodologije in ponuja več strategij za sestavljanje metodologije. Na začetku izberemo ali bomo izbrali obstoječo metodologijo kot osnovo in na njej naredili prilagoditve, ali pa bomo zgradili metodologijo na novo. Za tem določimo cilj, ki ga želimo doseči s konstruiranjem metodologij. Na podlagi cilja izberemo enega izmed treh pristopov za sestavljanje metodologije. To je pristop, ki temelji na sklopih, razširitvah ali paradigmah. Z izbranim pristopom nato sestavimo metodologijo. Ko je metodologija sestavljena, je kasneje še evalvirana.

Uporaba multikriterijskih tehnik

Kot nadgradnjo pristopa po sklopih, ki je bil predstavljen v okviru generičnega procesnega modela, je bil v delu [20] predstavljen izbor delov metodologij z multikriterijskimi tehnikami. To so tehnike za določanje uteži, kot npr. SMART (angl. Simple multiatribute technical rating) in outranking tehnika ELECTRE (angl. Elimination and choice corresponding to reality). Delovanje pristopa je prikazano na diagramu 2.5. Graf vsebuje dve glavni aktivnosti. To sta definiraj uteži in določi prioritete. Izbira med tema dvema aktivnostima je narejena na podlagi potrebe po uteževanju kriterijev. Uteževanje kriterijev omogoča analizo relativne pomembnosti. Ko kriteriji niso uteženi, to pomeni, da je njihova relativna pomembnost enaka. Uteževanje kriterijev lahko izvedemo s preprostim seštevanjem uteženih vrednosti, z izbiro kriterijev, ki morajo biti izboljšani najprej, s tehniko primerjave (angl. trade-off weighting) ali s analizo pomembnosti kriterijev. Uporabimo lahko več iteracij ali mehko uteževanje. Za določanje prioritet lahko uporabimo dva načina. To sta prioritetna strategija brez uteževanja ali prioritetna strategija z uteževanjem. Namen te faze je združitev alternativnih evalvacij v eno oceno za prioritizacijo alternativ. Pri prioritetni strategiji brez uteževanja lahko uporabimo tehniko



Slika 2.4: Generični proces za SME [18]

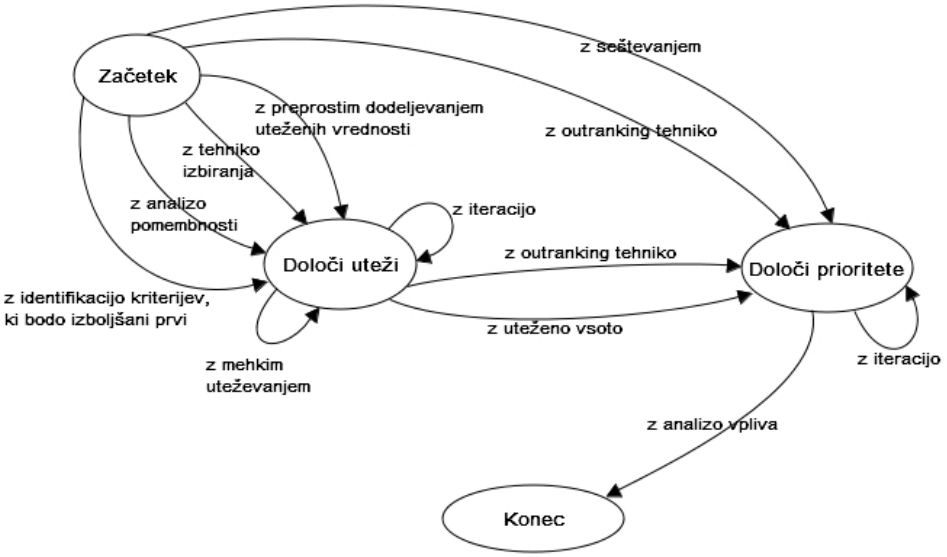
outranking brez uteževanja ali seštevanje ocen. Seštevanje ocen lahko izvedemo, če imajo vse ocene homogeno kvalitativno naravo in so lahko normalizirane. Tehnika outranking je lahko uporabljena za vse podatkovne tipe (kvalitativne in kvantitativne) in ne potrebuje normalizacije, vendar je najbolj zahtevna tehnika. Pri prioritetni strategiji z uteževanjem lahko izberemo med strategijama outranking z uteževanjem ali uteženo vsoto. Razlika med naštetima strategijama je podobna kot pri prioritetizaciji brez uteževanja.

Route map

Tehniko *route map* je predstavil van Slooten [21]. Ta uporablja dve vrsti gradnikov za situacijsko konstruiranje metodologij. Metodološki fragmenti (standardni in prilagojeni) in fragmenti načrta poti. Slednji se nanašajo na strategije, aktivnosti in produkte ter tudi na projektni management. *Route map* predstavlja potek poti skozi sistem. Uporablja se za prilagajanje splošnih metod v situacijsko prilagojene metode. Različne poti se uporabljajo za prilagoditev metod različnim situacijam.

Uporaba *deontic* matrik

Henderson-Sellers [22] in Graham [23] sta predstavila konstruiranje metodologij z uporabo *deontic* matrik. Te matrike so sedaj vgrajene v OPEN Process Framework (OPF). *Deontic*

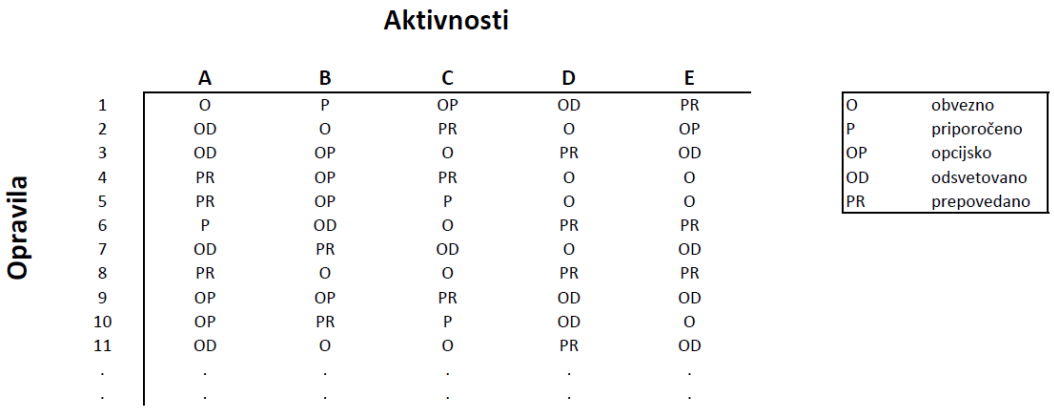


Slika 2.5: Diagram postopka izbora metod po multikriterijskih tehnikah

matrika je dvodimenzionalna matrika z vrednostmi, ki predstavljajo možnost uspešne zveze med vsakim parom fragmentov metodologij, izbranih iz repozitorija OPF. S tako matriko si lahko pomagamo pri kreiranju metodologije. Primer take matrike bi bila zveza aktivnosti in opravil, ki je prikazana na sliki 2.6. Za vsako aktivnost bi določili, kako pomembna je uporaba posameznega opravila (npr. obvezna, opcijska itd.). Nguyen [24] je identificiral sedem možnih *deontic* matrik: proces/aktivnost, aktivnost/opravilo, opravilo/tehnika, proizvajalec/opravilo, opravilo/produkt, proizvajalec/produkt in produkt/jezik. Na določitev vrednosti take matrike vpliva več faktorjev, npr. velikost projekta, organizacijska kultura, domena aplikacije, znanja in preference razvojne ekipe itd. Priporočeno je, da se za ocenjevanje uporablja pet nivojev. To so:

- obvezno,
- priporočeno,
- opcijsko,
- odsvetovano,
- prepovedano.

V praksi se velikokrat namesto petstopenjske lestvice uporabljajo kar binarne vrednosti. Ko je *deontic* matrika enkrat izpolnjena ponuja priporočila za izbor najbolj primernih



Aktivnosti

Opravila	A	B	C	D	E
1	O	P	OP	OD	PR
2	OD	O	PR	O	OP
3	OD	OP	O	PR	OD
4	PR	OP	PR	O	O
5	PR	OP	P	O	O
6	P	OD	O	PR	PR
7	OD	PR	OD	O	OD
8	PR	O	O	PR	PR
9	OP	OP	PR	OD	OD
10	OP	PR	P	OD	O
11	OD	O	O	PR	OD
.	.	.	.	.	.
.	.	.	.	.	.

O	obvezno
P	priporočeno
OP	opcijsko
OD	odsvetovano
PR	prepovedano

Slika 2.6: Primer *deontic* matrike

metod. Za večje organizacije je iz matrike lahko uporabnih več vrednosti. Na primer, namesto da določimo specifično tehniko za realizacijo opravila in prepovemo vse ostale, lahko razvojni ekipi omogočimo izbiro med eno ali več tehnikami, s katerimi so najbolj zadovoljni oz. so najbolj primerne za njihovo delo. Različne ekipe imajo lahko različne preference glede na njihova znanja.

Uporaba aktivnostnega diagrama UML

Aktivnostni diagrami UML so najbolj uporabni za prikaz poteka procesnega modela potem, ko so bili fragmenti že izbrani in sestavljeni [25]. Nekoliko manj podpore ponujajo izbiri fragmentov in sestavljanju fragmentov. Zato znanje skrbnika procesa igra večjo vlogo. Postopek razvoja nove metode s to tehniko je sestavljen iz naslednjih korakov:

- identifikacija potreb za novo metodo z analizo aplikacijskega konteksta,
- izbira metod, ki ustrezajo potrebnim vidikom, iz obstoječih metod,
- analiza izbranih metod in shranjevanje v bazo metod,
- združitev fragmentov metodologij v novo metodo za pridobitev situacijske metode.

Za modeliranje procesa ali dela procesa pri tem pristopu uporabljajo meta modelske tehnike. Ta tehnika je sestavljena iz dveh diagramov. To sta aktivnostni diagram UML in razredni diagram. Rezultat tega je nov hibridni diagram, ki se imenuje procesno-podatkovni diagram.

Meta-modelska tehnika

Pristop, ki ga je uporabil Brinkkemper [26] temelji na notaciji metodološkega fragmenta. Metodološki fragment je definiran kot opis metode razvoja programske opreme in njenih koherentnih delov. V primerjavi z elementom ISDM je metodološki fragment bolj podrobno klasificiran. Klasificiran je glede na tri dimenzije: perspektiva, stopnja abstrakcije in stopnja granulacije. Dimenzija perspektive združuje produktni in procesni fragment. Produktni fragment predstavlja končne izdelke, modele ali diagrame, procesni fragment pa prestavlja faze procesa, aktivnosti in opravila. Abstrakcijska dimenzija vsebuje konceptualni in tehnični nivo. Konceptualni fragmenti so opisi metod ali delov metod, tehnični fragmenti pa specifikacije implementacije operacijskih delov metod (npr. programska orodja). Stopnja granulacije pa predstavlja nivo dekompozicije v metodi. Iz procesnega vidika so lahko naprimer metode dekompozirane v faze, ki so v nadaljevanju razdeljene na aktivnosti in individualne korake. Za združitev metodoloških fragmentov je Brinkkemper predlagal nabor združitvenih pravil, ki so formalizirane v obliki predikatne logike prvega reda. Pravila povedo, kako povezati metodološke fragmente skozi nove nastale asociacije in koncepte. Pravila uvajajo omejitve v proces združevanja fragmentov in omogočajo skrbniku procesa pridobitev smiselnih metod. Večina omejitev je sintaktičnih, kljub temu pa poudarjajo tudi potrebno po semantičnih omejitvah. Ker semantične omejitve potrebujejo formalizacijo fragmentih semantik je Brinkkemper predlagal model v obliki ontologije, ki se imenuje MDM (angl. Methodology Data Model). Primer konceptov, ki so predlagani z MDM, so: aktivnost, cilj, skupina, problem, zahteva, vloga in sistem.

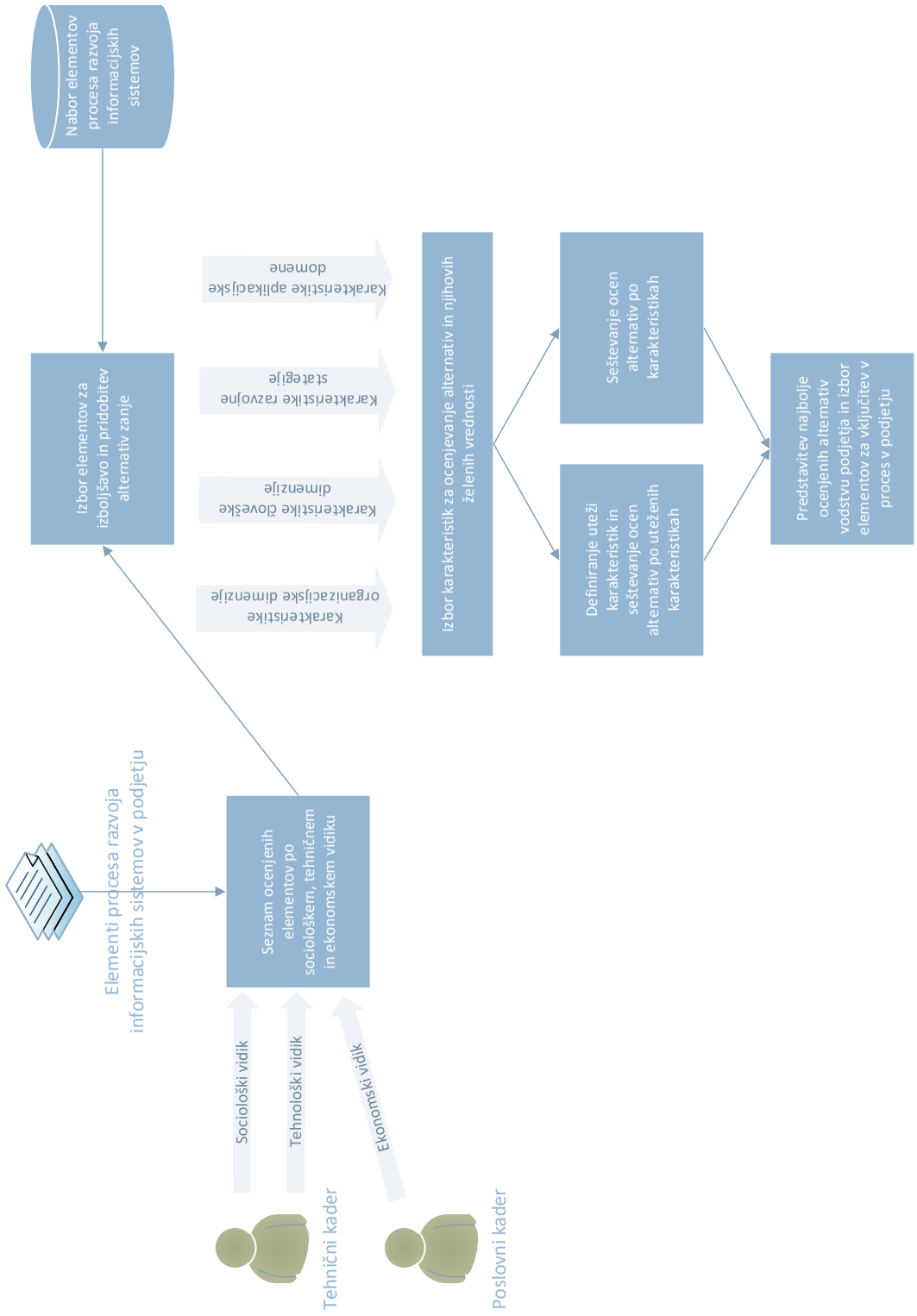
Poglavje 3

Opis ogrodja

V tem poglavju je predstavljena zgradba ogrodja. Ogrodje je sestavljeno iz več delov, ki so med seboj povezani in skupaj omogočajo izvedbo postopka izboljšave procesa razvoja informacijskih sistemov v podjetju. Vsak izmed delov ogrodja je v nadaljevanju podrobneje predstavljen in opisan. Deli ogrodja so:

- pripravljalni del,
- evalvacijski del,
- izbirni del,
- potrditveni del.

Slika 3.1 predstavlja zgradbo ogrodja.



Slika 3.1: Zgradba ogrodja

Predstavljen je način, kako v ogrodju od začetka postopka, preko posameznih delov ogrodja, pridobimo končni rezultat. Iz evalvacijskega dela vidimo, kako preko vprašalnikov, ki vsebuje sociološki, tehnološki in ekonomski vidik pridobimo seznam evalviranih elementov. S seznama potem izberemo elemente za ocenjevanje in pridobimo alternative zanje. Izbrani so tisti elementi, ki so bili slabo ocenjeni vsaj iz enega vidika. Za pridobitev alternativ je potreben repozitorij elementov označen z metapodakti. Zaradi prevelike obsežnosti vseh elementov, ki bi lahko bili v repozotoriju, in prevelike spremenljivosti elementov repozitorija v okviru magistrske naloge repozitorija nismo pripravili. Za elemente, ki smo jih ocenjevali, smo sproti pridobili možne alternative. V nadaljevanju iz nabora karakteristik, ki so predstavljene v poglavju 3.3.1, izberemo karakteristike, po katerih bomo ocenjevali alternative. Izbor se razlikuje za projekte in je odvisen od tega, po kakšnih kriterijih želimo pridobiti ocene za alternative. Alternative ocenimo po karakteristikah s seštevanjem ocen ali uteženo vsoto. Bolj podrobno je postopek predstavljen v poglavju 3.3.2. Na koncu alternative, ki so dobile najboljše ocene predstavimo vodstvu podjetja, ki se odloči, katere alternative so primerne za vključitev v proces dela v podjetju.

3.1 Pripravljalni del

Prvi del ogrodja je pripravljalni del. V tem delu najprej identificiramo elemente procesa razvoja informacijskih sistemov. V podjetju, kjer bomo uporabili ogrodje, skupaj s skrbnikom razvojnega procesa pregledamo, kako trenutno poteka razvojni proces in si zabeležimo elemente ISDM. Elemente lahko uvrstimo tudi v skupine. V isti skupini so elementi, ki nastopajo skupaj v določeni fazi razvoja informacijskih sistemov, kot je na primer analiza, načrtovanje itd. To nam kasneje pomaga pri evalvacijskem delu. Nato se lahko vzporedno izvajata aktivnosti specifikacije karakteristik razvojne ekipe ter priprava vprašalnikov in izbira deležnikov. Namen specifikacije karakteristik razvojne ekipe je pridobiti bolj podrobne informacije o razvojni ekipi, ki nam bodo pomagale pri izbirnem delu. Pri pripravi vprašalnikov za vsak prej identificiran element pripravimo vprašalnik za evalvacijo. Vprašalniki so predstavljeni v opisu evalvacijskega dela. V podjetju nato izberemo zaposlene, ki bodo izpolnjevali vprašalnike.

3.2 Evalvacijski del

Namen evalvacijskega dela je dobiti oceno o kakovosti trenutnega poteka procesa razvoja informacijskih sistemov v podjetju. Želimo si pridobiti obširen pogled na proces iz več vidikov. Sodelujoči v procesu ocenijo trenutni potek procesa s sociološkega, tehničnega in ekonomskega vidika. Za izvedbo tega dela ogrodja smo pripravili vprašalnike, preko

Pri <aktivnosti >sodelujem vsakič, ko se pri opravljanju mojega dela za to pojavi priložnost (te aktivnosti nikoli ne izpustim).
Zelo sem zadovoljen s/z <aktivnost >(ta aktivnosti ne bi mogla potekati bolje).

Tabela 3.1: Vprašanja za sociološki vidik

Moja razvojna skupina zelo učinkovito izvaja <aktivnost >(izvajanja te aktivnosti ne bi mogli izboljšati).
Naš pristop k <aktivnost >je v primerjavi z drugimi meni znanimi pristopi k tej aktivnosti zelo napreden.
Naš pristop k <aktivnost >je izmed vseh meni znanih pristopov s tega področja najbolj skladen z načinom dela moje razvojne skupine.

Tabela 3.2: Vprašanja za tehnološki vidik

katerih sodelujoči v procesu lahko podajo svoje ocene za posamezne vidike. Vprašalniki so sestavljeni z vprašanj po sedemstopenjski Likertovi lestvici (od 1 do 7). Uporabniki v procesu ISDM ocenijo elemente procesa iz tehničnega, sociološkega in ekonomskega vidika. Vsi uporabniki ne ocenjujejo procesa iz vseh vidikov, ampak samo tiste, za katere so ustrezno izobraženi. V nadaljevanju so v tabelah predstavljena vprašanja, ki jih vprašalnik vsebuje za posamezne vidike. V tabeli 3.1 so vprašanja za sociološki vidik, v tabeli 3.2 za tehnični vidik in v tabeli 3.3 za ekonomski vidik.

Po končanem ocenjevanju, ko zberemo ocene vseh uporabnikov, se začne analiza zbranih ocen. Elementi so glede na ocene uporabnikov razdeljeni v osem skupin:

- **Uporabni in ekonomsko učinkoviti elementi** so elementi, ki so s sociološkega, tehničnega in ekonomskega vidika dobili dobro oceno.

Obstoječ pristop k <aktivnost >močno niža stroške razvoja produkta.
Obstoječ pristop k <aktivnost >močno viša kakovost produkta.
Obstoječ pristop k <aktivnost >močno povečuje hitrost razvoja produkta.

Tabela 3.3: Vprašanja za ekonomski vidik

- **Uporabni in ekonomsko neučinkoviti elementi** so elementi, ki so s sociološkega in tehničnega vidika dobili dobro oceno ter slabo iz ekonomskega vidika.
- **Neuporabni in ekonomsko učinkoviti elementi** so elementi, ki so dobili slabo oceno iz tehničnega in sociološkega vidika ter dobro iz ekonomskega vidika.
- **Neuporabni in ekonomsko neučinkoviti elementi** so elementi, ki so s sociološkega, tehničnega in ekonomskega vidika dobili slabo oceno.
- **Neustrezni in ekonomsko učinkoviti elementi** so elementi, ki so dobili dobro oceno s sociološkega in ekonomskega vidika ter slabo iz tehničnega.
- **Neustrezni in ekonomsko neučinkoviti elementi** so elementi, ki so dobili dobro oceno s sociološkega vidika ter slabo iz tehničnega in ekonomskega vidika.
- **Nesprejeti in ekonomsko učinkoviti elementi** so elementi, ki so dobili dobro oceno iz tehničnega in ekonomskega vidika ter slabo s sociološkega vidika.
- **Nesprejeti in ekonomsko neučinkoviti elementi** so elementi, ki so dobili dobro oceno iz tehničnega vidika in slabo s sociološkega in ekonomskega vidika.

Pri izbiri elementov, za katere bomo iskali alternative, se osredotoča na elemente, ki so slabo ocenjeni vsaj z enega vidika. To so elementi, ki bodo v nadaljevanju izboljšani s pomočjo metod iz konstruiranja metodologij. Te izbrane elemente bomo podrobneje analizirali s projektnimi karakteristikami, ki so opisane v poglavju 3.3.1.

3.3 Izbirni del

Izbirni del je osrednji del ogrodja. Vhod v izbirni del je seznam evalviranih elementov ISDM, ki smo jih pridobili iz evalvacijskega dela. Elemente sortiramo po prioritetah. Višjo prioriteto imajo elementi, ki so slabše ocenjeni iz več vidikov. Glede na to, koliko elementov imamo, se odločimo, ali bomo obravnavali vse ali samo najvišje uvrščene po prioritetni lestvici. Namen tega koraka je pridobiti seznam elementov za izboljšavo, ki gredo v pregled vodstvu podjetja.

3.3.1 Specifikacija projektnih karakteristik

Ko izberemo primerne elemente, se lotimo specifikacije projektnih karakteristik. Projektne karakteristike povzete po delu [20] opisujejo glavne lastnosti projektov razvoja informacijskih sistemov. Projektne karakteristike vplivajo na izbor elementov. Vsak element je ocenjen na podlagi vpliva na posamezno karakteristiko. Skrbnik procesa oceni primernost elementov za posamezne dimenzije projektov.

Karakteristike izhajajo iz treh virov [20, 21, 27] in vključujejo štiri dimenzije:

Organizacijska dimenzija: organizacijska dimenzija predstavi organizacijski pogled razvoja informacijskih sistemov. Vključuje naslednje karakteristike: predanost vodstva podjetja, pomembnost, časovni pritisk, pomanjkanje virov, velikost in stopnja inovativnosti. Možne vrednosti in opisi karakteristik so prikazani v tabeli 3.4.

Človeška dimenzija: človeška dimenzija opisuje kvalitete ljudi, ki so vpleteni v proces razvoja informacijskih sistemov. Vključuje naslednje karakteristike: odpor in konflikti, strokovnost, jasnost in stabilnost ter vpletenost uporabnikov. Možne vrednosti in opisi karakteristik so prikazani v tabeli 3.5.

Aplikacijska domena: aplikacijska domena vsebuje formalnost, razmerja, odvisnosti, kompleksnost, aplikacijski tip, aplikacijsko tehnologijo, razdelitev projekta, ponovljivost, spremenljivost in spremenljivost artefaktov. Možne vrednosti in opisi karakteristik so prikazani v tabelah 3.6 in 3.7.

Razvojna strategija: razvojna strategija zadeva sistem programiranja, projektno organizacijo, strategijo realizacije, strategijo dostave, sledljivost projekta in število virov. Možne vrednosti in opisi karakteristik so prikazani v tabeli 3.8.

Karakteristika	**Opis**	**Možne vrednosti**	**Vir**
Predanost vodstva podjetja	V kolikšni meri vodstvo podjetja podpira projekt.	{nizka, normalna, visoka}	[21], [27], [20]
Pomembnost	Kako pomemben je projekt za organizacijo.	{nizka, normalna, visoka}	[21], [20]
Časovni pritisk	V kolikšni meri je predviden čas za projekt, ocenjen kot nezadosten.	{nizka, normalna, visoka}	[21], [20]
Pomanjkanje virov	V kolikšni meri je število finančnih, projektnih, začasnih in informacijskih virov za projekt ocenjeno kot nezadostno.	{nizka, normalna, visoka}	[21], [20]
Velikost	Velikost projekta.	{nizka, normalna, visoka}	[21], [27], [20]
Stopnja inovativnosti	Kako poslovno in tehnološko inovativen je projekt, glede na dosedanje projekte.	{nizka, normalna, visoka}	[21], [27], [20]

Tabela 3.4: Karakteristike organizacijske domene

Karakteristika	**Opis**	**Možne vrednosti**	**Vir**
Odpor in konflikti	V kolikšni meri imajo deležniki različne ali nasprotujoče si interese.	{nizka, normalna, visoka}	[21], [20]
Strokovnost	Kako dobro strokovno usposobljeni so testerji, razvijalci, oblikovalci in analitiki.	{nizka, normalna, visoka}	[21], [27], [20]
Jasnost in stabilnost	V kolikšni meri so cilji, potrebe in želje uporabnikov pri funkcionalnih zahtevah jasne in stabilne.	{nizka, normalna, visoka}	[21], [27], [20]
Vpletenost uporabnikov	Ali so uporabniki vpleteni realno ali virtualno.	{realna, virtualna}	[27], [20]

Tabela 3.5: Karakteristike človeške dimenzije

Karakteristika	**Opis**	**Možne vrednosti**	**Vir**
Formalnost	Kako formalna so pravila, procedure in standardi za poslovne procese in podporne informacije.	{nizka, normalna, visoka}	[21], [27], [20]
Razmerja	Kakšna so razmerja med novim informacijskim sistemom in ostalimi informacijskimi sistemi.	{nizka, normalna, visoka}	[21], [20]
Odvisnosti	V kolikšni meri je projekt odvisen od aktivnosti in pogojev zunaj projekta.	{nizka, normalna, visoka}	[21], [27], [20]
Kompleksnost	Kako kompleksne so funkcionalne komponente informacijskega sistema.	{nizka, normalna, visoka}	[21], [20]
Aplikacijski tip	Tip aplikacije, ki bo razvita.	{intraorganizacijska aplikacija, interogranizacijska aplikacija, organizacijsko-potrošniška aplikacija}	[27], [20]

Tabela 3.6: Karakteristike aplikacijske domene

Karakteristika	**Opis**	**Možne vrednosti**	**Vir**
Aplikacijska tehnologija	Uporabljena tehnologija pri razvoju aplikacije.	{aplikacija vsebuje podatkovno bazo, aplikacija je distribuirana, aplikacija vsebuje GUI}	[27], [20]
Razdelitev projekta	Kakšna je razdelitev projekta.	{en sam sistem, vzpostavljanje sistemsko orientiranih podprojektov, vzpostavljanje procesno orientiranih podprojektov, vzpostavljanje hibridnih podprojektov}	[21], [27], [20]
Ponovljivost	Ponovljivost razvoja aplikacije.	{nizka, normalna, visoka}	[20]
Spremenljivost	V kolikšni meri je aplikacija spremenljiva.	{nizka, normalna, visoka}	[20]
Spremenljivost artefaktov	V kakšni meri so spremenljivi artefakti.	{organizacijska, človeška, aplikacijska domena, razvojna strategija}	[20]

Tabela 3.7: Karakteristike aplikacijske domene

Karakteristika	**Opis**	**Možne vrednosti**	**Vir**
Vir sistema	Kateri obstoječi vir bo ponovno uporabljen pri razvoju.	{uporaba obstoječe kode, uporaba obstoječe funkcionalne domene, uporaba obstoječih vmesnikov}	[27], [20]
Projektna organizacija	Kakšna je projektna organizacija razvoja.	{standardna, prilagojena}	[27], [20]
Razvojna stategija	Uporabljena strategija razvoja sistema.	{outsourcing, iterativna, prototipiranje, po fazah, po delih}	[21], [27], [20]
Strategija realizacije	Strategija realizacije različnih podsistemov.	{naenkrat, inkrementalna, vzporedna, prekrivajoča}	[21], [27], [20]
Stategija dostave	Strategija dostave in uvajanja informacijskega sistema v organizacijo.	{naenkrat, inkrementalna, evolucijska}	[21], [27], [20]
Sledljivost projekta	V kolikšni meri lahko sledimo projektu.	{slaba, dobra}	[21], [27], (3)
Število ciljev	Koliko ciljev imamo pri razvoju projekta.	{en cilj, več ciljev}	[20]

Tabela 3.8: Karakteristike razvojne strategije

3.3.2 Izbor elementov na podlagi projektnih karakteristik

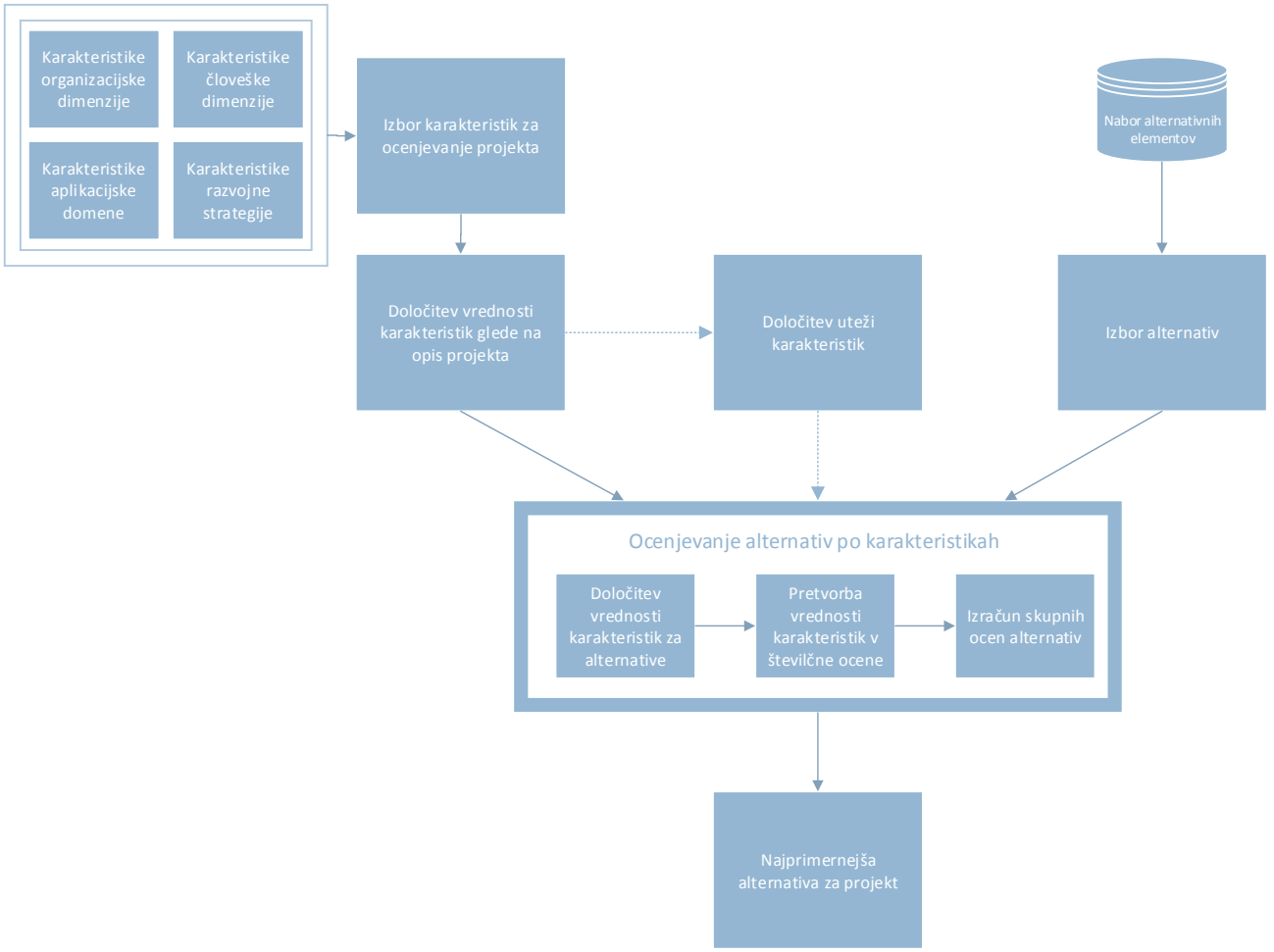
Izbor elementov temelji na postopku, ki je bil definiran v delu [20]. Postopek izbora elementov je prikazan na sliki 3.2. Za vsak element, za katerega bomo iskali alternative, izberemo, po katerih karakteristikah ga bomo ocenjevali. Izbiro naredi skrbnik procesa, ki določi, katere karakteristike so potrebne za projekt. Karakteristike, ki niso bile izbrane, se pri ocenjevanju ne upoštevajo. Nabor karakteristik je predstavljen v poglavju 3.3.1. Skrbnik procesa nato določi, kakšne so vrednosti izbranih projektnih karakteristik. Ko imamo specificirane projektne karakteristike, se lotimo izbora elementov, ki jih predlagamo kot izboljšave procesa razvoja informacijskih sistemov. Seznam alternativ je pridobljen iz repozitorija elementov, ki vsebuje elemente, označene z metapodatki. Zaradi prevelike obsežnosti nabora elementov ISDM in neprestanega spreminjanja elementov, v okviru magistrske naloge nismo ustvarili repozitorija elementov. Nabor elementov, ki smo jih uporabili v magistrski nalogi, smo zbrali le za elemente, ki jih v nadaljevanju ocenjujemo v študiji primera. Iz nabora vseh elementov iz repozitorija izberemo primerne elemente, ki bi bili lahko alternativa ocenjevanemu elementu. Te alternative v nadaljevanju ocenimo. Ocene za karakteristike se določijo na podlagi tega, kako se ujemajo z vrednostmi, ki jih je predhodno določil skrbnik procesa.

Pretvorba vrednosti karakteristik v številčne vrednosti

Na podlagi [20] smo pripravili postopek izračuna ocen. Prvi korak pri ocenjevanju je, da opisne vrednosti karakteristik za posamezne elemente pretvorimo v številčne vrednosti. S tem omogočimo seštevanje ocen posameznih karakteristik v skupno oceno, ki jih lahko na koncu primerjamo.

Za karakteristike, ki imajo možne vrednosti {nizka, normalna, visoka}, določimo ocene na sledeč način:

- Če skrbnik določi, da je želena vrednost karakteristike visoka, potem dobi element, ki ima vrednost karakteristike visoka tri točke, element z vrednostjo normalna dve točki in element z vrednostjo nizka eno točko.
- Če skrbnik določi, da je želena vrednost karakteristike nizka, potem dobi element, ki ima vrednost karakteristike nizka tri točke, element z vrednostjo normalna dve točki in element z vrednostjo visoka eno točko.
- Če skrbnik določi, da je želena vrednost karakteristike normalna, potem dobi element, ki ima vrednost karakteristike normalna dve točki, element z vrednostjo nizka eno točko in element z vrednostjo visoka eno točko.



Slika 3.2: Izbor elementov na podlagi projektnih karakteristik

Za karakteristike, ki imajo več možnih vrednosti in je želena vrednost karakteristike le ena vrednost, določimo ocene tako, da dobi element eno točko, če ima vrednost karakteristike enako želeni vrednosti in nič točk, če ima vrednost karakteristike različno od želene vrednosti.

Za karakteristike, ki imajo več možnih vrednosti in je želenih vrednosti za karakteristiko več, določimo ocene po formuli 3.1. V tej formuli O predstavlja številčno oceno alternative za karakteristiko, n predstavlja število možnih vrednosti karakteristike, V_i ima vrednost 1, če je možna vrednost karakteristike na mestu i izbrana kot želena in vrednost 0, če ni izbrana kot želena, K_i pa ima vrednost 1, če alternativni element vsebuje vrednost na mestu i iz možnih vrednosti karakteristike in 0, če je ne vsebuje.

$$O = \frac{\sum_{i=1}^{n} V_i * K_i}{\sum_{i=1}^{n} V_i} \tag{3.1}$$

Izračun ocene za alternativo

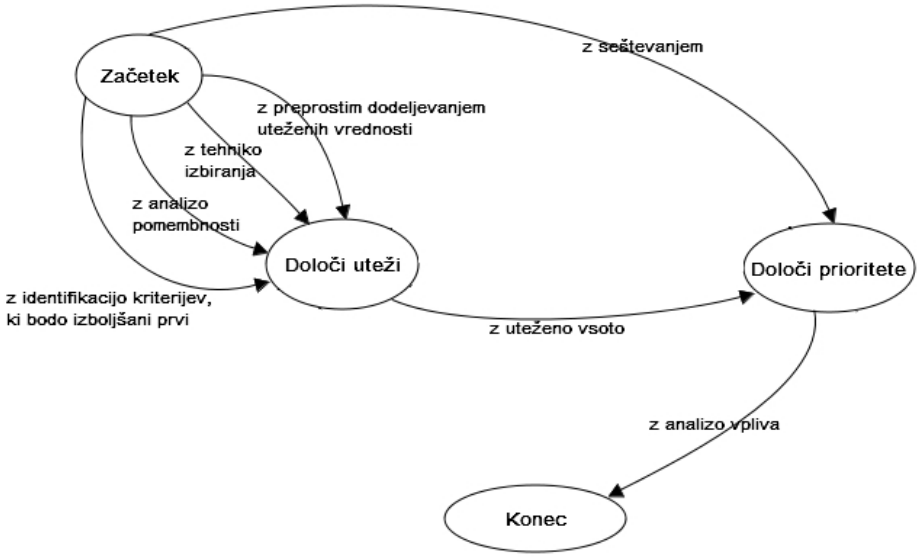
V nadaljevanju skupno oceno alternative izračunamo po formuli (3.2). V tej formuli X predstavlja skupno oceno alternative, m število karakteristik za ocenjevanje alternative, U_n utež posamezne karakteristike in O_n številčno oceno posamezne karakteristike za alternativo. Za uteževanje karakteristik se odločimo, ko želimo analizirati relativno pomembnost kriterijev. V primeru, da predpostavimo, da so vse karakteristike enako pomembne, potem vsem karakteristikam določimo utež U_n z vrednostjo 1.

$$X = \sum_{n=1}^{m} U_n * O_n \tag{3.2}$$

Tehnike uteževanja

Za izbor uteži lahko uporabimo več različnih tehnik, kot prikazuje slika 3.3. Te tehnike uteževanja predlagajo v delu [20]. Tehnike so:

- preprosto dodeljevanje uteženih vrednosti,
- identifikacija kriterijev, ki bodo izboljšani prvi (SWING) [28],
- tehnika izbiranja [29],
- analiza pomembnosti (angl. Simple Multiattribute Technical Rating - SMART) [30].



Slika 3.3: Diagram postopka izbora elementov

Tehnika SMART vsebuje naslednje korake: razvrstitev kriterijev glede na njihovo pomembnost, dodeljevanje vrednosti kriterijev od 1 do 100 in izračun relativne pomembnosti posameznega kriterija. Pri tehniki SWING so vsi kriteriji razumljeni kot slabi. Ekspert izbere tisti kriterij, ki mora biti izboljšan prvi in mu dodeli vrednost 100. Ista operacija je izvedena na ostalih kriterijih za določanje njihovih vrednosti. Povsod se določi kriterij, ki mora biti izboljšan prvi. Pri tehniki izbiranja odločevalec primerja dve alternativi glede na dva kriterija. Ostali kriteriji so nepomembni. Uteži teh dveh kriterijev so določene tako, da imata vrednosti uteži za alternativi, enako pomembnost za odločevalca. Operacija se ponavlja, dokler niso določene vse uteži. Tehniko SMART izberemo v primeru, da dobro poznamo vrednosti kriterijev in jih znamo primerjati med seboj. V primeru, da ne znamo dobro relativno primerjati vrednosti kriterijev, je priporočljivo uporabiti tehniko SWING. Tehniko izbiranja pa uporabimo, ko želimo pridobiti uteži s postopno primerjavo po dveh kriterijih.

3.4 Potrditveni del

Seznam elementov, ki so predlogi za izboljšavo procesa razvoja informacijskega sistema, iz našega ogrodja, damo na koncu v dokončno presojo še vodstvu podjetja. Vodstvu podjetja

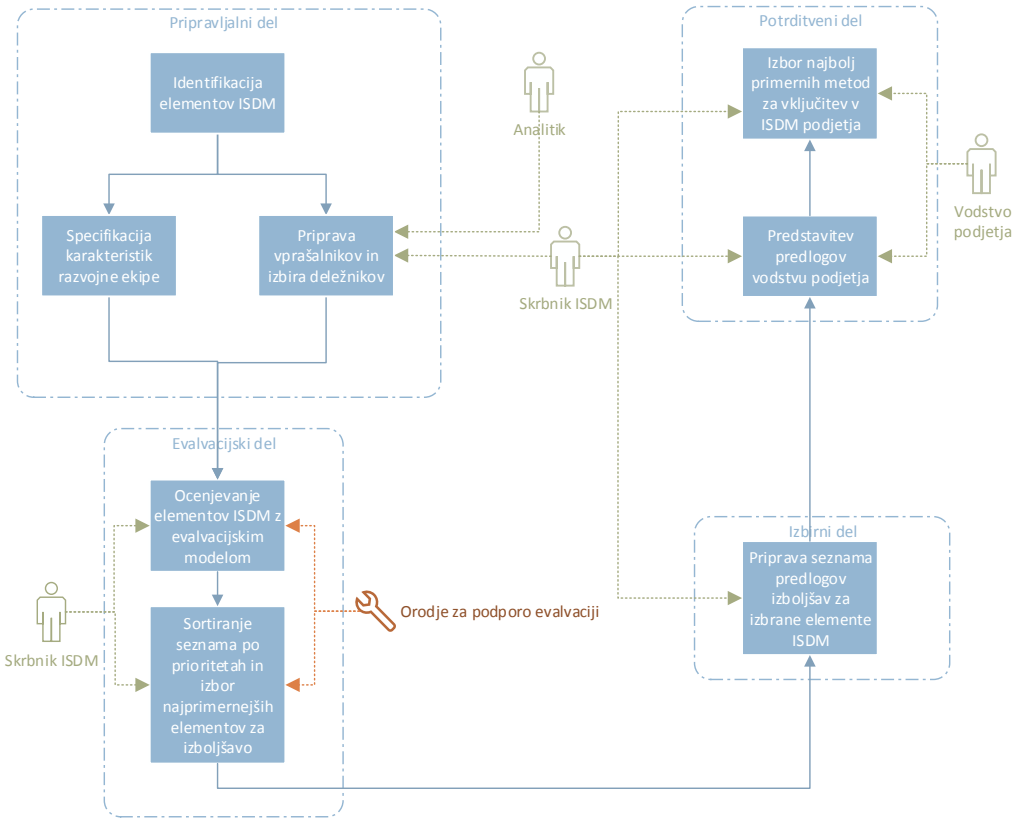
predstavimo, katere elemente ISDM smo analizirali. Pokažemo tudi rezultate evalvacije elementov. Predstavimo, med katerimi alternativami smo izbirali pri pripravi pristopov za izboljšavo procesa razvoja informacijskih sistemov. Nato predstavimo izbrane predloge izboljšav in razložimo, zakaj smo jih izbrali.

Seznam, ki ga dobijo iz našega ogrodja, jim svetuje, katere dele procesa razvoja informacijskih sistemov bi bilo dobro izboljšati in kako. Na podlagi njihovih odzivov je sestavljen končni seznam izbranih elementov, ki jih bo podjetje vpeljavo v njihov proces razvoja informacijskih sistemov.

Na podlagi odzivov podjetja lahko preverimo učinkovitost ogrodja. Ugotovimo, ali so rezultati ogrodja za podjetje relevantni in ali so z uporabo ogrodja pridobili koristne predloge izboljšav, ki jih bodo vpeljali v razvojni proces.

3.5 Postopek uporabe ogrodja

Na sliki 3.4 je predstavljen postopek uporabe ogrodja. Postopek se začne z identifikacijo elementov ISDM. V podjetju, kjer bomo uporabili ogrodje, pregledamo obstoječi proces razvoja informacijskih sistemov in si zapišemo identificirane elemente procesa razvoja informacijskih sistemov. Nato se lahko vzporedno začneta aktivnosti specifikacija karakteristik razvojne ekipe in priprava vprašalnikov in izbira deležnikov. V prvi aktivnosti specificiramo karakteristike razvojne ekipe, kot so velikost ekipe, izkušnje posameznikov v ekipi itd. Pri pripravi vprašalnikov si pomagamo z orodjem za podporo evalvaciji. Orodje je pripravljen vprašalnik v storitvi Google Forms. Vprašalnik prilagodimo tako, da mu dodamo imena elementov, ki jih bomo evalvirali. Prav tako moramo izbrati, kateri deležniki bodo izpolnjevali katere vprašalnike. To je odvisno od delovnega mesta, ki ga deležniki opravljajo. Na podlagi tega izpolnjuje tehnični kader s sociološkega in tehničnega vidika, poslovni kader iz ekonomskega vidika ter projektni vodje iz vseh treh vidikov. Po tem sledi faza ocenjevanja elementov ISDM z evalvacijskim modelom. S tem pridobimo ocene za posamezne elemente ISDM. Pridobljen seznam ocenjenih elementov sortiramo po prioritetah in izberemo najprimernejše elemente za izboljšavo. To naredimo na podlagi ocen elementov. Običajno izberemo elemente, ki so slabo ocenjeni iz več vidikov. Upoštevati pa moramo tudi medsebojne odvisnosti elementov in spremenljivost elementov. Na izbranem seznamu elementov za izboljšavo uporabimo naše ogrodje za pripravo predlogov izboljšav na podlagi evalvacije. Ta nam vrne seznam predlogov izboljšav za izbrane elemente ISDM. Seznam predlogov nato predstavimo vodstvu podjetja. To storimo z organizacijo sestanka, kjer predstavimo ugotovitve, do katerih smo prišli. Nato skupaj z vodstvom podjetja izberemo najbolj primerne metode za vključitev v ISDM podjetja. S tem se zaključi postopek uporabe ogrodja.



Slika 3.4: Proces uporabe ogrodja z prikazom orodij za podporo posameznim aktivnostim

Poglavje 4

Študija primera

Z namenom preizkusa delovanja ogrodja smo izvedli študijo primera v podjetju. V nadaljevanju je predstavljen opis podjetja, na katerem smo izvedli študijo primera. Opisan je trenutni proces razvoja informacijskih sistemov v podjetju. Na koncu so predstavljeni še rezultati študije primera. Prikazani in razloženi so rezultati evalvacijskega dela, izbirnega dela in potrditvenega dela.

4.1 Opis podjetja

Študijo primera smo izvedli v podjetju, ki se ukvarja z informatiko v logistiki. V podjetju je zaposlenih 25 ljudi. Podjetje zagotavlja celovite in kakovostne programske rešitve na področju notranje logistike in avtomatiziranih skladišč. Razvijajo informacijske sisteme za upravljanje skladišč (angl. warehouse management systems - WMS). Podjetje se ukvarja s/z:

- analizo in svetovanjem pri informacijski podpori in upravljanju skladišč,
- razvojem programskega paketa za upravljanje skladišč,
- vzdrževanjem in podporo naročnikom,
- podporo avtomatskemu skladiščenju (angl. material flow system - MFS),
- podporo mobilnim terminalom v ročnem skladiščenju z uporabo črtne kode in radiofrekvenčne identifikacije,
- razvojem branžnih rešitev za kovinske, farmacevtske in prehranske izdelke,
- svetovanjem pri prenovi poslovnih procesov v logistiki in pri uvedbi informacijske podpore zanje.

4.2 Trenuten proces razvoja informacijskih sistemov

Podjetje izvaja komercialne in interne projekte. Projekti zahtevajo prilagajanje standardne verzije informacijskega sistema za upravljanje skladišč in nadgradnjo s funkcionalnostmi za konkretno stranko. Projektni vodja dnevno spremlja izvajanje nalog in praviloma enkrat tedensko preveri realizacijo projekta in o tem poroča direktorju oz. stranki. Postopek izvajanja vseh aktivnosti je v celoti podprt z aplikacijo za upravljanje opravil JIRA. Vsak modul je verzioniran, kar omogoča orodje za verzioniranje kode Perforce. Spremembe med posameznimi verzijami so vidne v orodju JIRA in zapisih v orodju za sodelovanje med ekipami Confluence. Tehnična dokumentacija je obvladovana v orodju Confluence in vsebuje kratek povzetek sprememb med verzijami in povezavo na orodje JIRA, kjer so podrobnejši opisi na ravni posameznih opravil.

V nadaljevanju je proces opisan v več sklopih. V vsakem sklopu so opisane aktivnosti, ki se izvajajo znotraj določenega dela procesa razvoja informacijskih sistemov.

4.2.1 Analiza in načrt

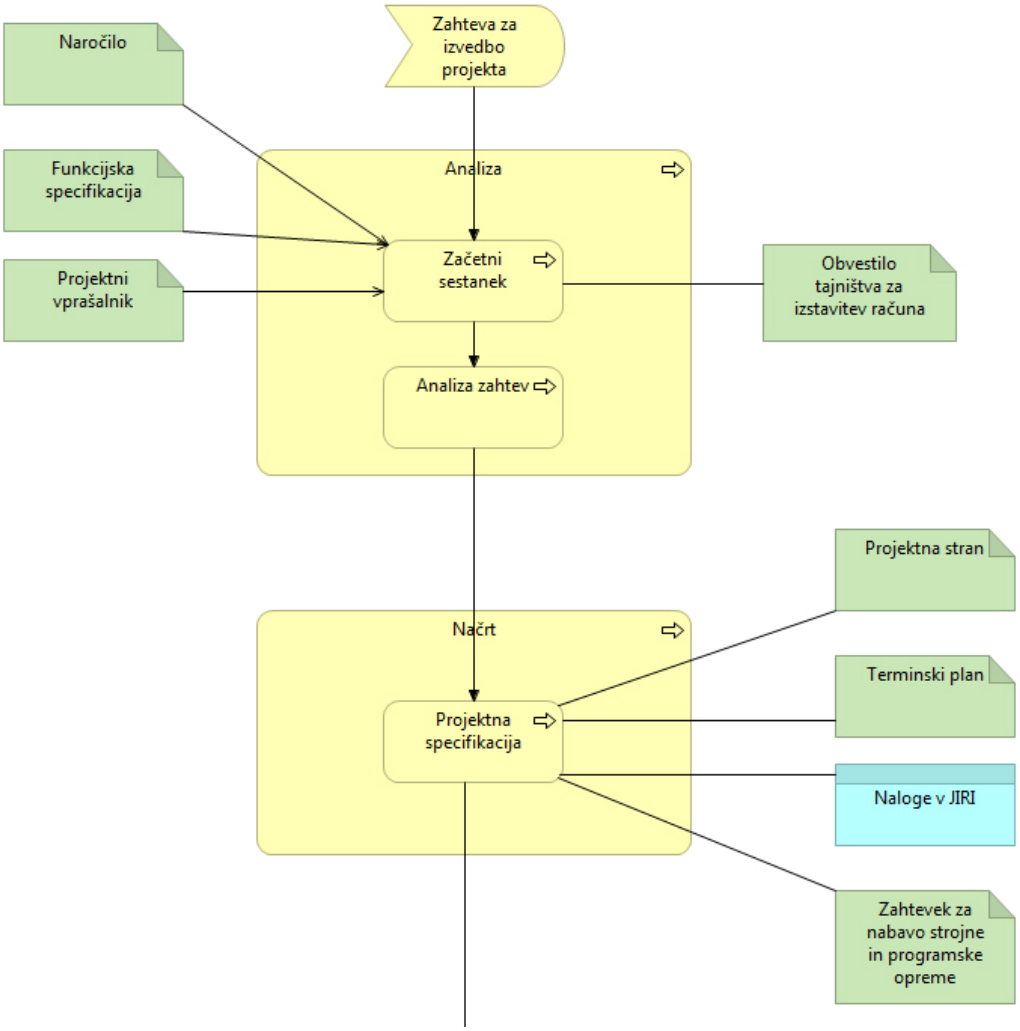
Vsak projekt v podjetju se začne z uvodnim sestankom. Na njem se zberejo vsi zaposleni, ki bodo pri projektu sodelovali. Bistvo sestanka je pregled naročila in funkcijske specifikacije. Na podlagi tega, kaj je opredeljenega v sklopu naročila in funkcijske specifikacije se začne izpolnjevati projektni vprašalnik. Izpolni se, kaj je treba v okviru projekta izdelati, kakšni so časovni roki in projektne specifike. Skozi funkcijsko specifikacijo se analizira zahtevnost projekta in tipi opravil, ki jih bo treba v okviru projekta izdelati. Na podlagi sestanka se pripravi obvestilo tajništva za izstavitev računa, kjer so podani časovni roki in finančna zgradba projekta. Po začetnem sestanku sledi opravilo analiza zahtev. Zahteve, ki so bile analizirane iz naročila in funkcijske specifikacije, se zapiše v obliki seznama. Ugotovi se, kakšne so zahteve za razvoj programske opreme. Analizira se tudi, če obstajajo konfliktne zahteve med različnimi deležniki. Za analizo zahtev sledi kreiranje projektne specifikacije. Za projekt se ustvari projektna stran v orodju Confluence. Na njej se beležijo tedenski zapisniki projekta in diagrami, ki prikazujejo, ali projekt poteka v predvidenih časovnih okvirjih. Poleg projektne strani se pripravi tudi terminski plan. V njem v grobem specificiramo, do kdaj bo zaključena posamezna faza projekta. Za tem se izdelajo naloge v sistemu JIRA. Za to poskrbi vodja projekta, ki naloge podrobno opiše in primerno razporedi. Glede na analizirane zahteve za projekt se napiše tudi zahtevek za nabavo strojne in programske opreme. Potek aktivnosti analiza in načrt je prikazan na sliki 4.1. V tabeli 4.1 so našteti posamezni koraki aktivnosti.

Ime koraka	Opis
Začetni sestanek	Ekipa, ki bo delala na projektu, se zbere in pregleda naročilo in funkcijsko specifikacijo projekta. Izpolnijo tudi projektni vprašalnik.
Analiza zahtev	Analizirajo se zahteve, ki so v specifikaciji projekta.
Projektna specifikacija	Napiše se projekta specifikacija, ki vsebuje projektno stran, terminski plan, naloge v sistemu za opravila in zahtevek za nabavo strojne in programske opreme.

Tabela 4.1: Koraki aktivnosti analiza in načrt

4.2.2 Implementacija

Naslednja aktivnost je implementacija, ki je prikazana na sliki 4.2. Prvi korak aktivnosti je priprava razvojnega okolja. Razvijalci si pripravijo razvojno okolje na svojih računalnikih. V okviru priprave razvojnega okolja si namestijo orodje za urejanje programske kode in orodje za upravljanje s podatkovno bazo. S strežnika za kontrolo verzij si prenesejo zadnjo verzijo programske opreme. Poskrbijo tudi za povezavo na testne strežnike. Za tem, ko imajo razvijalci pripravljeno razvojno okolje, lahko začnejo izvajanje nalog. Za dodeljevanje nalog razvijalcem je zadolžen projektni vodja, ki poskrbi, da so vse naloge ustrezno razporejene med razvijalci. Naloge se dodelijo preko orodja JIRA. Ko razvijalci dobijo nalogo, se lotijo programiranja naloge. Programiranje poteka v programskih jezikih Java, Javascript in PL/SQL. Vzporedno s programiranjem se izvajata tudi opravili priprava produkcijskega okolja in izdelava uporabniškega priročnika. Pri pripravi produkcijskega okolja se poskrbi za konfiguracijo produkcijskega strežnika, povezavo VPN do produkcijskega strežnika, omrežne nastavitve in poskrbi se za ustrezno izvajanje varnostnih kopij. Uporabniški priročnik se izdeluje sproti, ko se programira določene funkcionalnosti sistema. V uporabniškem priročniku je uporaba funkcionalnosti sistema opisana na nivoju, da jo znajo razumeti končni uporabniki. Po uspešni implementaciji programske kode za



Slika 4.1: Potek aktivnosti analiza in načrt

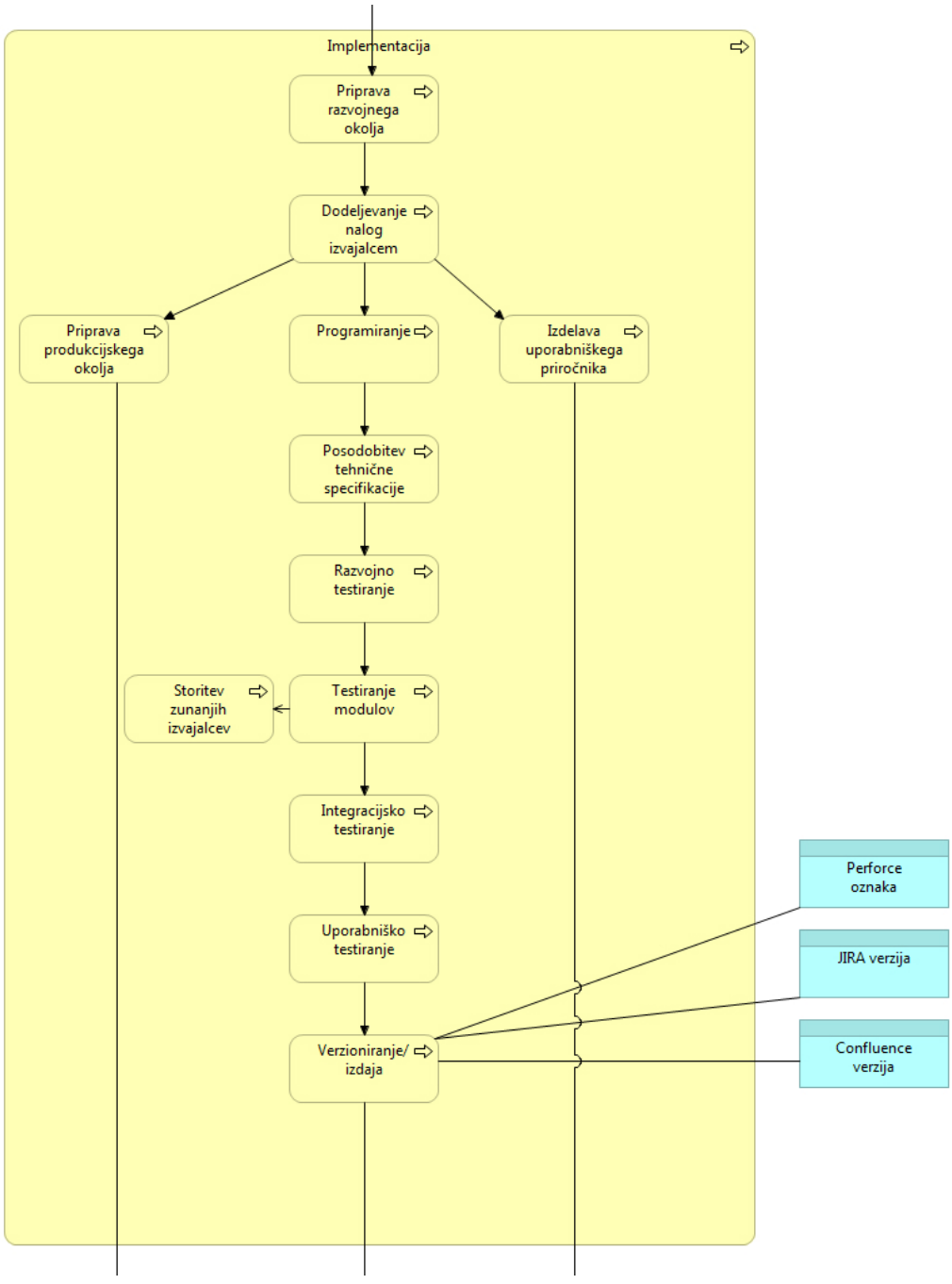
nalogo razvijalci prenesejo kodo v sistem za verzioniranje kode Perforce. Med programiranjem lahko pride do ugotovitev, da bo treba spremeniti tehnično dokumentacijo. V skladu z ugotovitvami se tehnična specifikacija projekta lahko spremeni tekom implementacije. Po uspešni implementaciji programske kode je kodo treba testirati. To je izvedeno v okviru opravila razvojno testiranje. Razvijalci glede na zahteve opravila testirajo svojo kodo in preverjajo pravilnost delovanja kode za različne primere. Po uspešnem razvojnem testiranju se izvaja testiranje modulov. S tem preverjamo, da ob spreminjanju funkcionalnosti modula, modul še vedno ustrezno deluje. Če projekt vsebuje tudi storitve zunanjih izvajalcev, je treba testirati tudi delovanje modulov zunanjih izvajalcev. Po testiranju modulov se izvede še integracijsko testiranje. Pri integracijskem testiranju so posamezni moduli združeni in testirani kot skupina. Za tem, ko je izvedeno razvojno testiranje, testiranje modulov in integracijsko testiranje je programska oprema dovolj dobra, da se lahko začnejo izvajati uporabniški testi. Uporabniki dobijo verzijo programske opreme z določenimi funkcionalnostmi in so pozvani, da izvajajo in poročajo rezultate o testiranju funkcionalnosti, za katere so zadolženi. Ko je enkrat tudi uporabniško testiranje uspešno zaključeno, se naredi verzija programske opreme. Ta vključuje oznako v sistemu za verzioniranje kode Perforce, verzijo v sistemu za upravljanje opravil JIRA in verzijo v orodju za sodelovanje med ekipami Confluence. Koraki aktivnosti implementacija so našteti v tabeli 4.2

4.2.3 Zagon

Za implementacijo sledi zagon, ki je prikazan na sliki 4.3. Pri zagonu se pri naročniku projekta namesti končna verzija programske opreme, ki je bila razvita in testirana. Delovanje programske opreme se še dodatno testira na produkcijskem sistemu. Ko je verzija programske opreme dovolj testirana, se izvede zagon produkcijskega sistema. Zagnan sistem za upravljanje skladišča se v nadaljevanju pregleda. Preveri se, da so izpolnjene projektne naloge iz kontrolnega seznama. Ko so vse projektne naloge iz kontrolnega seznama uspešno opravljene sledi zaključek zagona. Pri zaključku zagona se pripravi tehnična dokumentacija programske opreme, certifikati šolanja uporabnikov in uporabniški priročnik. S tem stranka pridobi vse potrebne dokumente, da lahko sama začne uporabljati programsko opremo. Kasneje se posodobi še projektna dokumentacija. Projektno dokumentacijo se posodobi z ugotovitvami, do katerih smo prišli tekom zagona. Hkrati se posodobi tudi projektna stran v orodju Confluence. Dopolni se jo z informacijami, ki smo jih pridobili tekom zagona in ostalimi potrebni informacijami, ki bodo potrebne pri vzdrževanju projekta. Uspešno zagnan projekt se bolj podrobno spremlja še dva meseca. Med tem časom se odpravljajo morebitne odprte točke, do katerih je prišlo med uporabo sistema. Koraki

Ime koraka	**Opis**
Priprava razvojnega okolja	Razvijalci pripravijo razvojno okolje. Pripravijo programsko kodo, konfigurirajo podatkovno bazo itd.
Dodeljevanje nalog razvijalcem	Preko sistema za upravljanje opravil JIRA se dodelijo naloge razvijalcem.
Priprava produkcijskega okolja	Priprava strežnika, kjer bo nameščen projekt in vzpostavitev povezave VPN do tega strežnika ter zagotovitev varnostnega kopiranja podatkov.
Programiranje	Razvoj programske kode za projekt.
Izdelava uporabniškega priročnika	Izdelava navodil za uporabo programa.
Posodobitev tehnične specifikacije	Dopolnitev tehnične specifikacije z ugotovitvami, ki so bile ugotovljene med fazo programiranja.
Razvojno testiranje	Razvijalci sproti testirajo programsko kodo med razvojem.
Testiranje modulov	Testiranje delovanje modulov.
Integracijsko testiranje	Integracijsko testiranje med seboj povezanih opravil.
Uporabniško testiranje	Informacijski sistem je testiran s strani izbranih uporabnikov.
Verzioniranje/izdaja	Označitev končne verzije programske kode projekta.

Tabela 4.2: Koraki aktivnosti implementacija



Slika 4.2: Potek aktivnosti implementacija

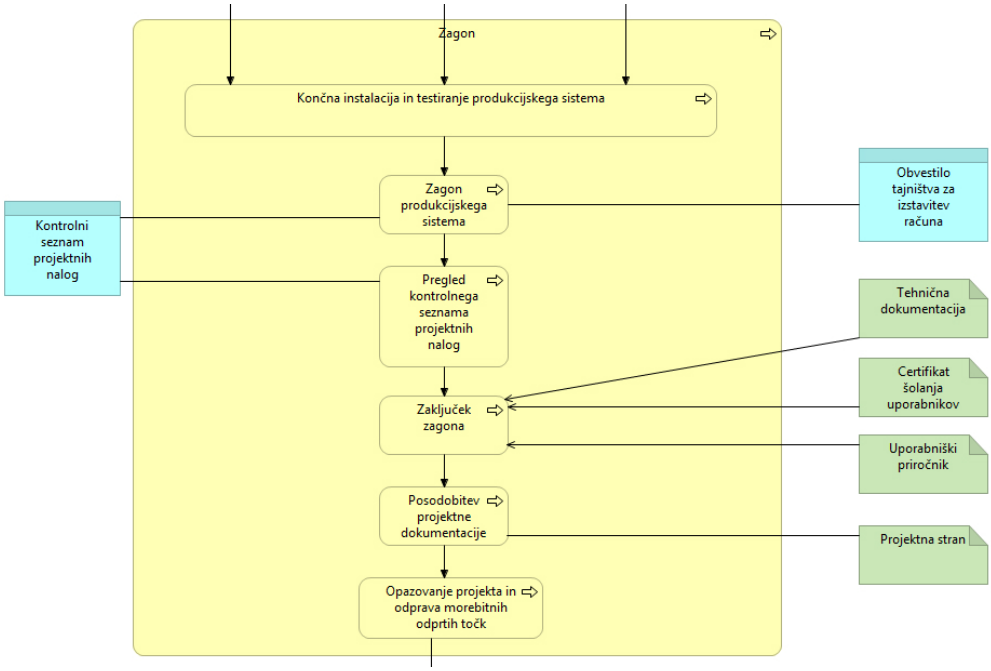
Ime koraka	**Opis**
Končna instalacija in testiranje produkcijskega sistema	Na produkcijski strežnik se namesti končna instalacija projekta. Izvede se testiranje informacijskega sistema na produkcijskem sistemu.
Zagon produkcijskega sistema	Izvede se zagon produkcijskega sistema pri naročniku.
Pregled kontrolnega seznama projektnih nalog	Na kontrolnem seznamu so opravila, ki jih je treba preveriti ob vsakem zagonu. Preveri se, če so bila vsa opravila uspešno izvedena.
Zaključek zagona	Ko je ugotovljeno, da je bil projekt uspešno zagnan, se pripravi tehnična dokumentacija, certifikate šolanja uporabnikov in uporabniški priročnik.
Posodobitev projektne dokumentacije	Posodobi se projekta stran projekta.
Opazovanje projekta in odprava morebitnih odprtih točk	Dva meseca po zagonu se projekta bolj podrobno spremlja in odpravlja morebitne odprte točke.

Tabela 4.3: Koraki aktivnosti zagon

aktivnosti zagon so našteti v tabeli 4.3.

4.2.4 Vzdrževanje

Zadnja faza je vzdrževanje, ki je prikazana na sliki 4.4. Pred začetkom vzdrževanja se izvede zaključevanje projekta. Tu gre predvsem za pregled projekta s strani oddelka za podporo. Ti preverijo seznam pogojev, ki jih mora projekt vsebovati, da ga lahko oni uspešno prevzamejo v vzdrževanje. Projekt se dopolni do te mere, da so vsi pogoji iz tega seznama izpolnjeni. Za tem sledi postimplementacijski sestanek. Ta sestanek organizira projektni vodja, ki povabi projektno ekipo, direktorja, tehničnega direktorja, vodjo oddelka za podporo, predstavnika za kakovost in smiselno še druge sodelavce. Med sestankom se



Slika 4.3: Potek aktivnosti zagon

piše zapisnik, ki mora vsebovati naslednje točke:

- razmerje med prodajno vrednostjo, interno oceno potrebnega dela in stroškov ter dejansko izvedenim delom in dejanskimi stroški,
- pravočasnost izvedbe in časovna odstopanja od začetnega projektnega plana,
- informacije o tem ali je bil obseg projekta v celoti realiziran in komentar morebitnih odstopanj,
- ocena kakovosti izvedbe in zadovoljstva stranke,
- informacije o dodatno prodanih storitvah,
- informacije o naročenih licencah za bazo Oracle in druge informacije o licencah in morebitni strojni opremi,
- pozitivne in negativne izkušnje na projektu,
- smernice, ki se jih držimo na prihodnjih projektih,
- ukrepi, ki jih moramo izvesti,
- informacije o novih funkcionalnostih in preverba ali je bil izveden prenos znanja o novih funkcionalnostih podpori uporabnikom in implementaciji.

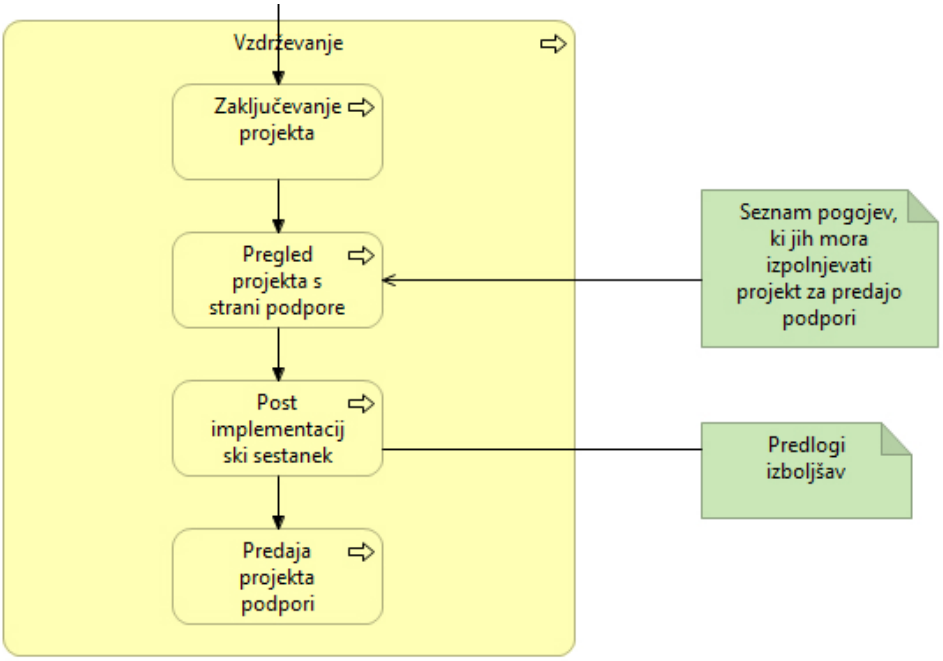
Ime koraka	**Opis**
Zaključevanje projekta	Opravi se še zadnja potrebna opravila na projektu.
Pregled projekta s strani podpore	Oddelek podpore v podjetju pregleda projekt in poda svoje mnenje o projektu.
Postimplementacijski sestanek	Ekipa, ki je delala na projektu, se zbere in poda predloge izboljšav za prihodnje projekte.
Predaja projekta podpori	Zaključen projekt se preda oddelku podpore v podjetju, ki skrbi za pomoč uporabnikom in odpravo morebitnih težav.

Tabela 4.4: Koraki aktivnosti vzdrževanje

Po izvedbi postimplementacijskega sestanka se projekt preda v vzdrževanje oddelku za podporo. Ta kasneje zagotavlja podporo stranki in zagotavlja, da sistem vseskozi nemoteno deluje. Koraki aktivnosti vzdrževanje so predstavljeni v tabeli 4.4.

4.3 Izbor elementov za študijo primera

V pripravljalnem delu našega ogrodja najprej identificiramo elemente ISDM. To smo storili na podlagi korakov aktivnosti pri trenutnem poteku razvoja informacijskih sistemov v podjetju, ki smo jih opisali v prejšnjem poglavju. Določeni koraki aktivnosti so lahko direktno pretvorjeni v elemente, lahko pa tudi združimo več korakov v en element. Pri pripravi elementov moramo biti pozorni na to, da bodo elementi dovolj dobro opredeljeni, tako da si bodo uporabniki namen elementa predstavljali enako. Elementi, ki smo jih izbrali za študijo primera, so: začetni sestanek, analiza zahtev, projektna specifikacija, priprava razvojnega okolja, dodeljevanje nalog razvijalcem, priprava produkcijskega okolja, programiranje, razvojno testiranje, testiranje modulov, integracijsko testiranje, uporabniško testiranje, končna instalacija in testiranje produkcijskega okolja, zagon produkcijskega okolja, zaključek zagona, pregled projekta s strani podpore in postimeplementacijski sestanek.



Slika 4.4: Potek aktivnosti vzdrževanje

To so elementi, ki izhajajo iz prej opisanih korakov aktivnosti. Za študijo primera smo jih izbrali, ker menimo, da predstavljajo ključne korake pri poteku posameznih aktivnosti in so dovolj dobro opredeljeni, da si jih bodo uporabniki enako razlagali.

4.4 Rezultati študije primera

Rezultati študije primera so prikazani po posameznih delih ogrodja. Najprej so predstavljeni rezultati evalvacijskega dela, nato rezultati izbirnega dela in na koncu rezultati potrditvenega dela.

4.4.1 Rezultati evalvacijskega dela

Evalvacija procesa razvoja informacijskih sistemov je potekala preko spletnih anket. Ankete so bile izdelane v spletnem orodju Google Forms. Orodje omogoča izdelavo dobro strukturiranih anket. Za izdelavo študije primera smo izdelali tri vrste anket. To so anketa za ocenjevanje ekonomskega vidika, anketa za ocenjevanje sociološkega in tehničnega vidika ter anketa za ocenjevanje vseh vidikov. Zaposleni v podjetju so ocenjevali elemente procesa razvoja informacijskih sistemov, ki so navedeni v tabeli 4.5.

Številka elementa	**Ime elementa**
0	Začetni sestanek
1	Analiza zahtev
2	Projektna specifikacija
3	Priprava razvojnega okolja
4	Dodeljevanje nalog razvijalcem
5	Priprava produkcijskega okolja
6	Programiranje
7	Razvojno testiranje
8	Testiranje modulov
9	Integracijsko testiranje
10	Uporabniško testiranje
11	Končna instalacija in testiranje produkcijskega okolja
12	Zagon produkcijskega okolja
13	Zaključek zagona
14	Pregled projekta s strani podpore
15	Postimplementacijski sestanek

Tabela 4.5: Ocenjevani elementi procesa razvoja informacijskih sistemov

Po končanem ocenjevanju smo začeli z analizo rezultatov. Za vsakega uporabnika smo za vsak element sešteli ocene za vsa vprašanja za posamezen vidik. Tako smo za vsakega uporabnika dobili za vsak element po eno oceno za posamezen vidik. Nato smo izračunali povprečne ocene elementov za posamezen vidik. To smo naredili tako, da smo sešteli ocene vseh uporabnikov za posamezen vidik in jih delili s številom uporabnikov, ki so ocenjevali element. Hkrati smo izračunali tudi povprečne ocene vseh elementov za posamezen vidik. S tem smo pridobili pozicije osi diagramov, ki so predstavljeni v nadaljevanju. Rezultati povprečij in standardnih odklonov ocen elementov po posameznih vidikih so prikazani v tabeli 4.6. Opazimo, da imajo v povprečju elementi najvišjo oceno s sociološkega vidika, malo nižjo iz tehničnega vidika in najnižjo z ekonomskega vi-

dika. Vrednosti za standardne odklone ocen so precej visoke za nekatere vidike elementov začetni sestanek, projektna specifikacija, priprava razvojnega okolja, dodeljevanje nalog razvijalcem, priprava produkcijskega okolja, programiranje, razvojno testiranje, testiranje modulov, zagon produkcijskega okolja, pregled projekta s strani podpore in postimplementacijski sestanek. V nadaljevanju smo tem elementov posvetili posebno pozornost in s pomočjo podrobnejše analize ugotavljali, kaj je razlog za to.

Slika 4.5 prikazuje evalvacijo elementov po sociološkem in tehničnem vidiku. Na sliki opazimo, da je bilo pet elementov ocenjenih kot sociološko sprejetih in tehnično učinkovitih. En element je bil ocenjen kot sociološko nesprejet in tehnično neučinkovit, pet elementov kot tehnično učinkovitih in sociološko nesprejetih, pet elementov, pa je bilo tako s sociološkega kot tehničnega vidika ocenjenih slabo. Oba elementa iz skupine vzdrževanje sta bila ocenjena kot nesprejeta. To kaže na to, da se vzdrževanje tehnično izvaja dovolj učinkovito, vendar z izvajanjem teh aktivnosti uporabniki niso najbolj zadovoljni. Treba bi bilo izboljšati elemente s sociološkega vidika. Pri skupinah implementacija in testiranje opazimo, da sta iz vsake skupine dva elementa zelo dobro ocenjena iz tehničnega in sociološkega vidika, dva elementa pa sta iz obeh vidikov ocenjena podpovprečno. To nakazuje na to, da se aktivnosti implementacije in testiranja v celoti ne izvajata dovolj učinkovito in imata precej možnosti za izboljšave tako iz tehničnega kot sociološkega vidika. Elementi, ki bi jih bilo treba izboljšati, so, razvojno testiranje, testiranje modulov, dodeljevanje nalog razvijalcem in priprava produkcijskega okolja.

Na sliki 4.6 vidimo evalvacijo po ekonomskem in sociološkem vidiku. Na sliki je razvidno, da so bili trije elementi ocenjeni kot sociološko sprejeti in ekonomsko učinkoviti. Trije elementi so bili ocenjeni kot sociološko sprejeti in ekonomsko neučinkoviti. Pet elementov je bilo ocenjenih kot ekonomsko neučinkovitih in sociološko nesprejetih. Kot ekonomsko učinkovitih in sociološko sprejetih, pa je bilo ocenjenih pet elementov. Oba elementa iz skupine vzdrževanje sta bila ekonomsko ocenjena podpovprečno. To nakazuje na to, da vodstvo ocenjuje, da izvajanje aktivnosti vzdrževanje podjetje stane preveč denarja. Najbolje ocenjeni elementi z ekonomskega vidika so iz skupine zagon. Vodstvo je zadovoljno z ekonomskim učinkom teh elementov. Prav tako pa so v večini primerov s sociološkega vidika z izvajanjem teh elementov zadovoljni tudi uporabniki.

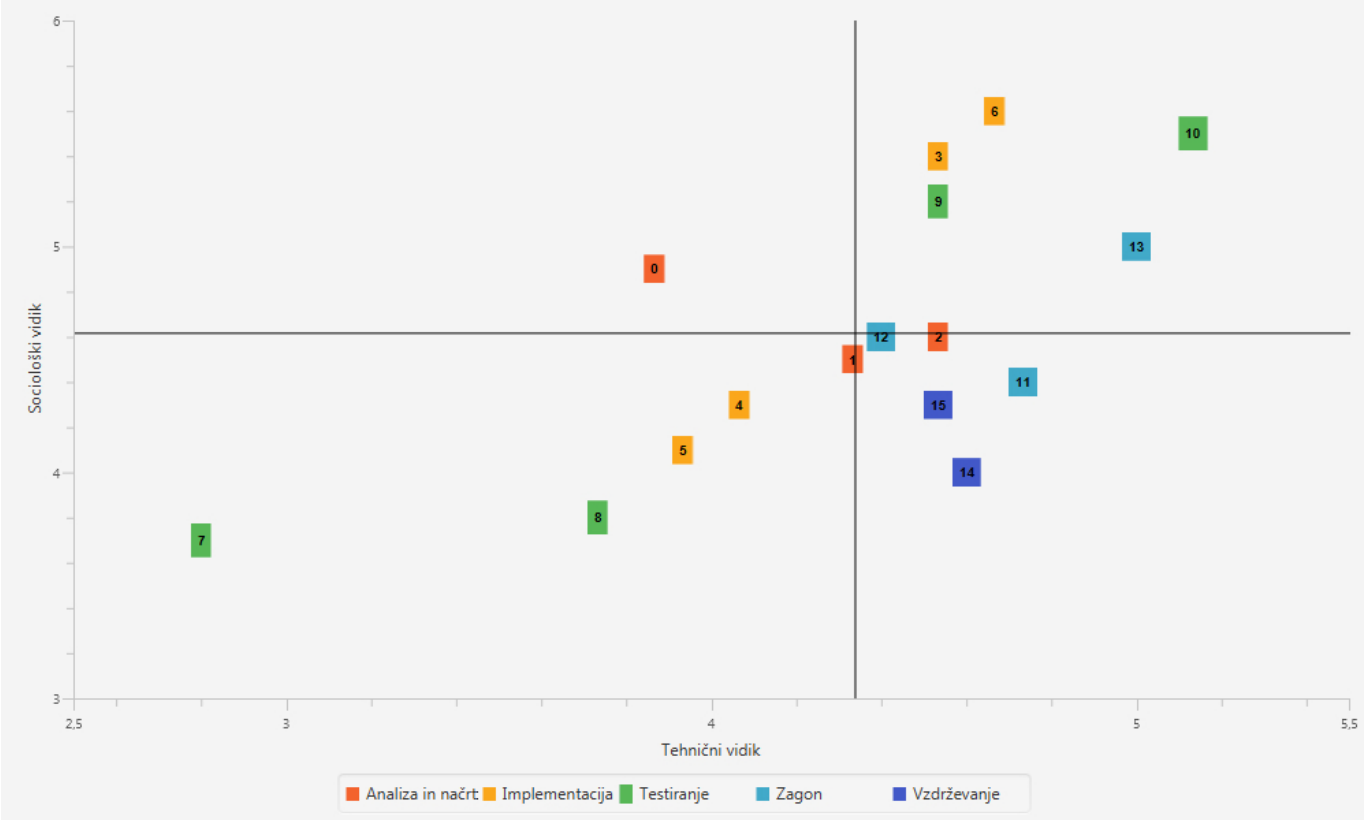
Slika 4.7 predstavlja evalvacijo po ekonomskem in tehničnem vidiku. Na sliki je razvidno, da je bilo kar šest elementov ocenjenih kot tehnično in ekonomsko učinkovitih. Iz tega je mogoče sklepati, da tehnično učinkoviti elementi pogosto pripomorejo k dobri ekonomski učinkovitosti. Štirje elementi so bili tehnično ocenjeni kot učinkoviti in ekonomsko kot neučinkoviti. Iz obeh vidikov so bili kot neučinkoviti ocenjeni štirje elementi. Dva elementa, pa sta bila ocenjena kot ekonomsko učinkovita in tehnično neučinkovita. Najbolje so se tako iz tehničnega kot ekonomskega vidika izkazali elementi iz skupine zagon. Aktivnost zagona se v podjetju izvaja tehnično in ekonomsko učinkovito. Oba elementa iz

Ime elementa	**Sociološki vidik**		**Tehnični vidik**		**Ekonomski vidik**	
	Povprečje	Odklon	Povprečje	Odklon	Povprečje	Odklon
Začetni sestanek	4.90	0.58	3.87	1.53	3.33	1.33
Analiza zahtev	4.50	0.84	4.33	0.73	4.50	0.50
Projektna specifikacija	4.60	1.24	4.53	1.09	4.50	0.50
Priprava razvojnega okolja	5.40	0.97	4.53	1.19	4.67	0.33
Dodeljevanje nalog razvijalcem	4.30	1.57	4.07	1.74	4.33	0.33
Priprava produkcijskega okolja	4.10	1.32	3.93	1.82	3.50	1.50
Programiranje	5.60	1.32	4.67	1.74	3.33	1.00
Razvojno testiranje	3.70	0.98	2.80	1.29	3.83	0.50
Testiranje modulov	3.80	0.98	3.73	1.10	4.17	0.83
Integracijsko testiranje	5.20	0.51	4.53	0.93	4.17	0.83
Uporabniško testiranje	5.50	0.45	5.13	0.65	4.83	0.83
Končna instalacija in testiranje produkcijskega okolja	4.40	0.97	4.73	1.06	4.33	0.67
Zagon produkcijskega okolja	4.60	1.53	4.40	1.02	4.83	0.83
Zaključek zagona	5.00	0.84	5.00	0.92	5.00	1.00
Pregled projekta s strani podpore	4.00	1.14	4.60	0.57	4.17	0.17
Postimplementacijski sestanek	4.30	1.91	4.53	0.86	3.33	1.33
Povprečje vsi elementi	**4.62**	**1.07**	**4.34**	**1.14**	**4.18**	**0.78**

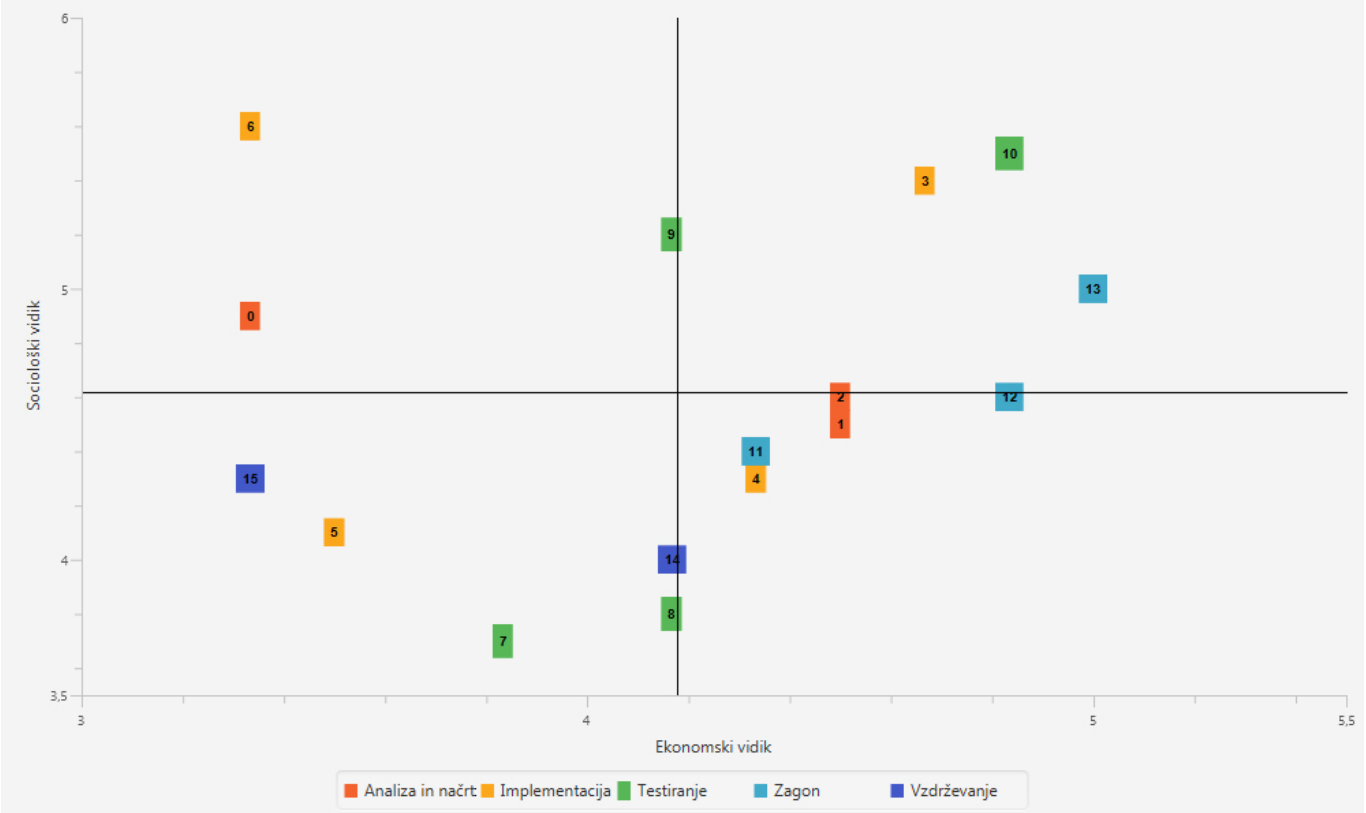
Tabela 4.6: Povprečja in standardni odkloni ocen za posamezni vidik

skupine vzdrževanje sta bila uvrščene med tehnično učinkovite in ekonomsko neučinkovite. Aktivnost vzdrževanja programske opreme se v podjetju izvaja tehnično dovolj učinkovito in ekonomsko premalo učinkovito.

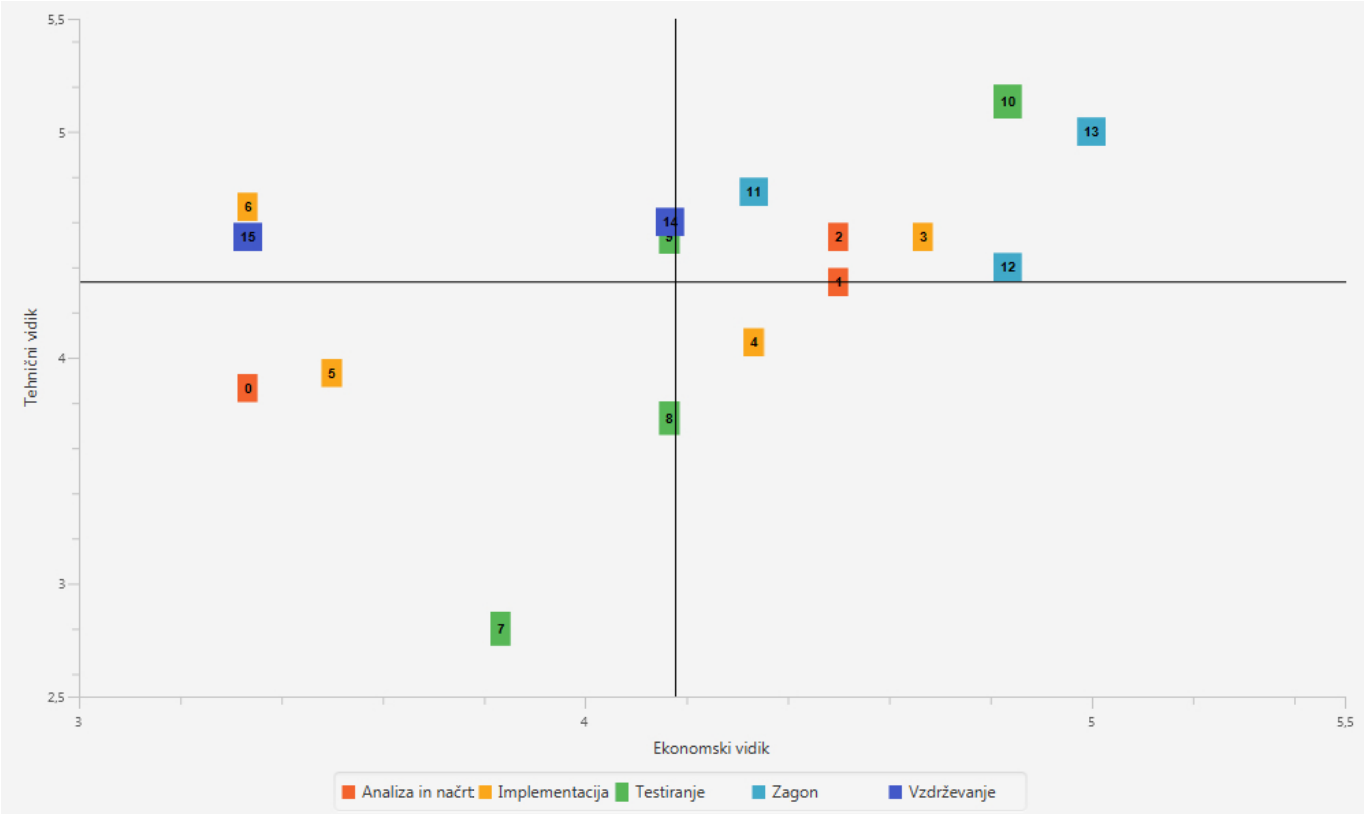
Slika 4.8 prikazuje ocene po vseh treh vidikih. Na osi x je ocenjen tehnični vidik in na osi y sociološki vidik. Rdeča barva predstavlja elemente, ki so bili z ekonomskega vidika ocenjeni pod povprečjem, zeleno pa so obarvani tisti elementi, ki so bili z ekonomskega vidika nad povprečjem. Vidimo, da je bila polovica elementov ocenjena kot ekonomsko učinkovito in polovica kot ekonomsko neučinkovitih. Elementi, ki so bili z ekonomskega vidika ocenjeni nad povprečjem, se nahajajo veliko bližje skupini uporabnih elementov kot elementi, ki so bili z ekonomskega vidika ocenjeni podpovprečno.



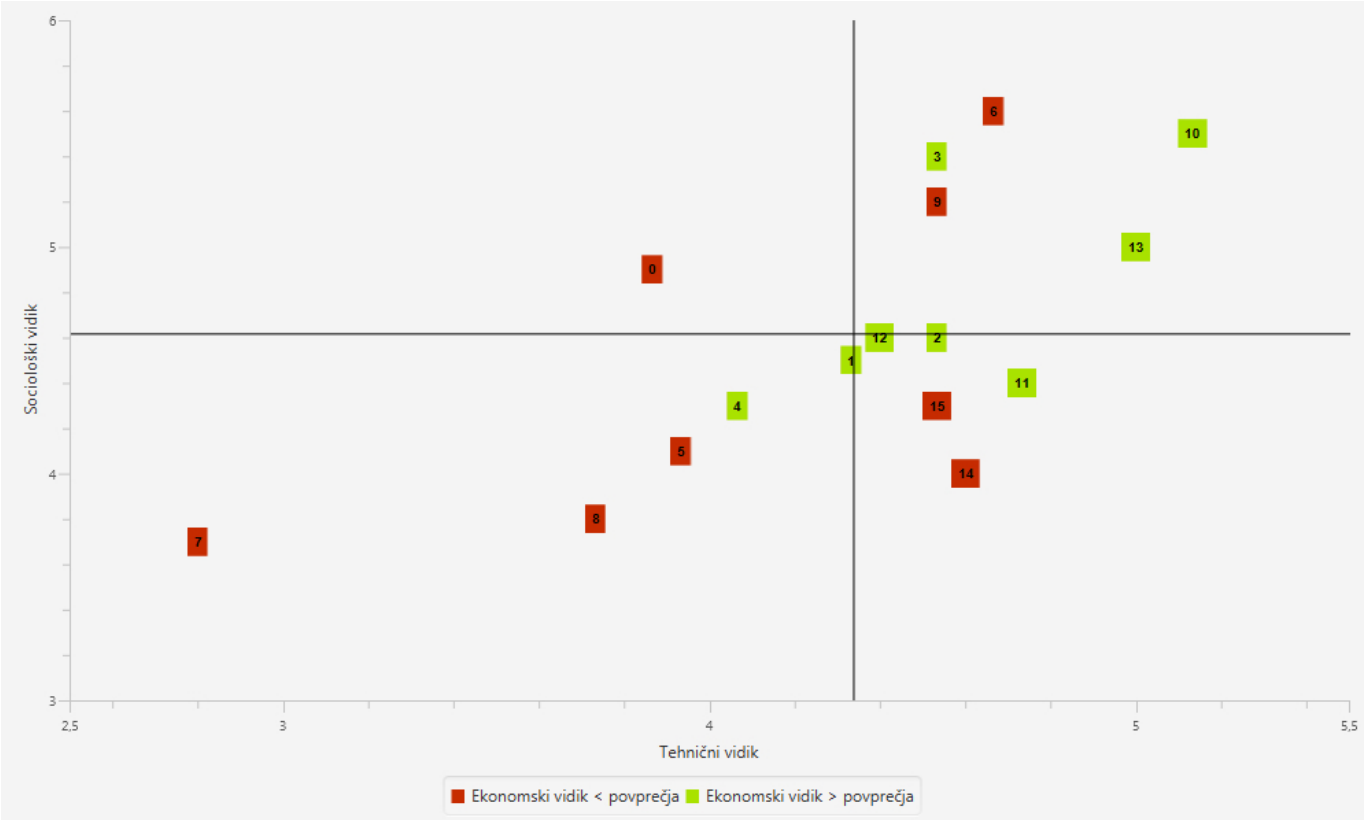
Slika 4.5: Evalvacijski diagram s tehničnim in sociološkim vidikom



Slika 4.6: Evalvacijski diagram z ekonomskim in sociološkim vidikom



Slika 4.7: Evalvacijski diagram z ekonomskih in tehničnih vidikom



Slika 4.8: Evalvacijski diagram z vsemi tremi vidiki

V nadaljevanju je vsak posamezni element analiziran s sociološkega, tehničnega in ekonomskega vidika. Analiza elementov se nanaša na razsevne diagrame, ki so bili pred tem našteti.

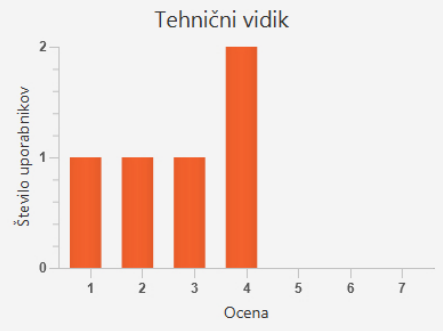
- **0 - začetni sestanek**: element je bil uvrščen med neučinkovite elemente. To pomeni, da so ga uporabniki dobro ocenili s sociološkega vidika in slabo iz tehničnega. Iz diagramov, ki vključujeta tudi ekonomski vidik, pa ugotovimo, da tudi ekonomsko ni dovolj dober, ker je povsod ekonomsko podpovprečjem.
- **1 - analiza zahtev**: element analiza zahtev je iz tehničnega vidika umeščen ravno na povprečje vidika. S sociološkega vidika, pa je malo pod povprečjem. Ker je iz obeh vidikov blizu povprečja pomeni, da bi potreboval manjše izboljšave tako s sociološkega kot iz tehničnega vidika. Iz ekonomskega vidika, pa je bil element dobro ocenjen.
- **2 - projektna specifikacija**: element projektna specifikacija je bil uvrščen med uporabne elemente. To pomeni, da ne bi potreboval nobenih izboljšav. Ampak ker je s sociološkega vidika na meji povprečja, bi ga bilo s tega vidika treba nekoliko izboljšati. Z ekonomskega vidika je bil dobro ocenjen.
- **3 - priprava razvojnega okolja**: ta element je bil uvrščen med uporabne elemente. Je dovolj dobro tehnično učinkovit, dobro sprejet med uporabniki in tudi ekonomsko učinkovit.
- **4 - dodeljevanje nalog razvijalcem**: element je bil uvrščen med neuporabne elemente. Tako s sociološkega kot tehničnega vidika se nahaja malo pod povprečjem, zato potrebuje izboljšave z obeh vidikov. Ekonomsko pa je bil ocenjen kot dovolj učinkovit.
- **5 - priprava produkcijskega okolja**: ta element je bil tako s sociološkega kot tehničnega vidika malo pod povprečjem. Potrebuje manjše izboljšave iz obeh vidikov. Z ekonomskega vidika je precej pod povprečjem, zato s tega vidika potrebuje večje izboljšave.
- **6 - programiranje**: element je bil dobro ocenjen tako s sociološkega vidika kot tehničnega vidika, zato je uvrščen v skupino uporabnih elementov. Potreboval pa bi izboljšave z ekonomskega vidika, kjer je bil podpovprečno ocenjen.
- **7 - razvojno testiranje**: element je bil izmed vseh elementov najslabše ocenjen tako s sociološkega kot tehničnega vidika. Z ekonomskega vidika je malo pod povprečjem. Element potrebuje večje izboljšave s sociološkega in tehničnega vidika ter manjše z ekonomskega.

- **8 - testiranje modulov**: element je bil uvrščen med neuporabne elemente. Potreboval bi izboljšave tako s sociološkega kot tehničnega vidika. Z ekonomskega vidika je uvrščen ravno na mejo povprečja, tako da bi potreboval tudi manjše ekonomske izboljšave.
- **9 - integracijsko testiranje**: uporabniki so element dobro ocenili, tako s sociološkega kot tehničnega vidika. Element je uvrščen med uporabne elemente. Z ekonomskega vidika je bil uvrščen na mejo povprečja, tako da bi tu potreboval manjše izboljšave.
- **10 - uporabniško testiranje**: element je bil med najbolje ocenjenimi elementi tako s sociološkega kot tehničnega vidika. Prav tako je bil dobro ocenjen tudi z ekonomskega vidika. Element zaenkrat ne potrebuje nobenih izboljšav.
- **11 - končna instalacija in testiranje produkcijskega okolja**: element je bil uvrščen med nesprejete elemente. To pomeni, da je tehnično dovolj učinkovit, vendar ga uporabniki neradi uporabljajo. Potreboval bi manjše izboljšave, da bi bil med uporabniki bolje sprejet. Z ekonomskega vidika je bil ocenjen kot učinkovit.
- **12 - zagon produkcijskega okolja**: element je tako s sociološkega kot tehničnega vidika uvrščen blizu povprečja. Iz obeh vidikov potrebuje manjše izboljšave. Ekonomsko je bil ocenjen kot dovolj učinkovit.
- **13 - zaključek zagona**: element je bil iz vseh vidikov zelo dobro ocenjen. Trenutno ne potrebuje izboljšav.
- **14 - pregled projekta s strani podpore**: ta element je bil uvrščen med nesprejete elemente. Tehnično je dovolj dober, vendar ga uporabniki niso dobro sprejeli. S sociološkega vidika potrebuje izboljšave. Z ekonomskega vidika je uvrščen ravno na mejo povprečja, zato tudi iz tega vidika potrebuje manjše izboljšave.
- **15 - postimplementacijski sestanek**: element je uvrščen med neučinkovite elemente. Potrebuje nekaj izboljšav s sociološkega vidika. Z ekonomskega vidika je bil ocenjen kot zelo neučinkovit in bi potreboval večje izboljšave iz tega vidika.

4.4.2 Rezultati izbirnega dela

Po končanem evalvacijskem delu smo dobili ocene za sociološki, tehnološki in ekonomski vidik za vse elemente procesa razvoja informacijskih sistemov, ki smo jih identificirali v študiji primera. Za prikaz izbirnega dela smo si izbrali tri elemente, ki so bili podpovprečno ocenjeni vsaj z enega vidika. To so elementi razvojno testiranje, analiza zahtev in dodeljevanje nalog razvijalcem. Ti elementi so v nadaljevanju podrobneje analizirani.

Slika 4.9: Sociološki vidik



Slika 4.10: Tehnični vidik

Razvojno testiranje

Element razvojno testiranje je bil najslabše ocenjen tako s sociološkega kot tehničnega vidika. Prav tako je bil tudi ekonomsko ocenjen podpovprečno. Na sliki 4.9 vidimo porazdelitev ocen za vprašanje s sociološkega vidika in na sliki 4.10 porazdelitev za vprašanje iz tehničnega vidika. Vidimo, da so ocene večine uporabnikov zelo nizke. Razlog za slabe ocene elementov je nedefiniran postopek izvajanja razvojnega testiranja. Razvijalci sproti testirajo programsko kodo, vsak s svojim pristopom k testiranju. Zaradi obsežnosti testiranja njihovi pristopi niso primerni za vsa testiranja in so zato podali slabše ocene iz tehnološkega in sociološkega vidika. Vodstvo ocenjuje, da je čas, ki ga razvijalci porabijo za razvojno testiranje predlog, zato so podali slabšo oceno z ekonomskega vidika.

Kot alternative za element razvojno testiranje smo identificirali štiri elemente. To so regresijsko testiranje, sprejemno testiranje, kontinuirano testiranje in destruktivno testiranje. Alternative so opisane v tabeli 4.7.

Ime alternative	**Opis alternative**	**Vključitev v proces podjetja**
Regresijsko testiranje	Tip testiranja, kjer preverjamo, da se prejšna razvita in testirana programska oprema še vedno pravilno izvaja, kljub spremembam.	Vključitev bi bila možna kot nadgradnja razvojnega testiranja z uporabo orodja za stalno integracijo (angl. continuous integration).
Sprejemno testiranje	Preverjamo, da so zahteve iz specifikacije izpolnjene.	Razvojno testiranje bi razširili s tem, da bi za projekt pripravili seznam specifikacij, katere bi med razvojnim testiranjem preverjali, da so izpolnjene.
Kontinuirano testiranje	Izvajamo avtomatske teste, kot del procesa razvoja programske opreme.	Za projekt bi definirali avtomatske teste, ki preverjajo osnovne zahteve projekta. Med razvojnim testiranjem bi se izvedli tudi avtomatski testi, kar bi izboljšalo to aktivnost.
Destruktivno testiranje	Preverjamo, da programska oprema še vedno deluje kljub vnosu nepravilnih ali nepričakovanih vnosov.	Pri razvojnem testiranju bi dali več poudarka tudi na preverjanje delovanja pri nepričakovanih vnosih. To bi lahko izvedli s testiranjem enot (angl. unit testing).

Tabela 4.7: Opis alternativ razvojnega testiranja

Projektne karakteristike, po katerih bomo ocenjevali alternative, so iz organizacijske dimenzije stopnja inovativnosti, iz človeške dimenzije strokovnost, iz aplikacijske domene formalnost, kompleksnost in ponovljivost ter iz razvojne strategije, strategija realizacije. Za uteževanje karakteristik se nismo odločili, ker ocenjujemo, da so vse karakteristike enako pomembne.

Glede na opis projekta in evalvacijo so bile identificirane vrednosti projektnih karakteristik. To so:

- stopnja inovativnosti: visoka,
- strokovnost: nizka,
- formalnost: visoka,
- kompleksnost: nizka,
- ponovljivost: visoka,
- strategija realizacije: inkrementalna, vzporedna, prekrivajoča.

Namen projekta je razvoj informacijskega sistema za upravljanje skladišča. Za projekt smo ocenili, da je tehnično in poslovno zelo inovativen, glede na dosedanje projekte. Kljub temu pa kompleksnost funkcionalnih komponent ni visoka. Zaradi tehničnih inovativnosti, ki jih projekt vsebuje, smo ocenili, da je strokovna usposobljenost kadrov nizka. Formalnost standardov za poslovne procese je bila ocenjena kot visoka. Zaradi tega bo tudi ponovljivost razvoja projekta visoka. Strategije realizacije različnih podsistemov bodo potekale inkrementalno, vzporedno in prekrivajoče.

Stopnje projektnih karakteristik za alternative smo določili na podlagi opisov alternativ in so prikazane v tabeli 4.8. Za regresijsko testiranje smo ocenili, da prinaša najmanj inovativnosti in potrebne strokovnosti kadrov, ostale alternative, pa smo ocenili z normalno stopnjo inovativnosti in strokovnosti. Najbolj formalni sta alternativi sprejemno in kontinuirano testiranje. Regresijsko in destruktivno testiranje sta bila ocenjena z nizko stopnjo formalnosti. Zaradi višje formalnosti je tudi kompleksnost sprejemnega in kontinuiranega testiranja višja kot pri regresijskem in destruktivnem testiranju. Ponovljivost vseh alternativ razen destruktivnega testiranja ocenjujemo kot visoko. Destruktivnemu testiranju smo dodelili normalno stopnjo ponovljivosti, ker pri tem testiranju uporabljamo nepričakovane vnose, zaradi katerih je ponovljivost testiranja malo manjša. Sprejemno, kontinuirano in destruktivno testiranje smo ocenili kot primerne za testiranje podsistemov, ki so razviti inkrementalno, vzporedno ali prekrivajoče. Regresijsko testiranje pa smo ocenili kot primerno za testiranje podsistemov, ki so razviti inkrementalno. Za podsisteme, ki bodo razviti vzporedno ali prekrivajoče to testiranje ne ponuja dovolj podpore. Ocene so bile dodeljene po postopku, ki je opisan v poglavju 3.3.2. Po seštetih ocenah, ki so prikazane v tabeli 4.9, je bila kot najboljša alternativa izbrano kontinuirano testiranje.

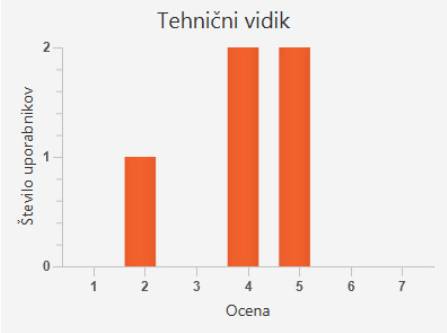
Alternativa	**Stopnja inovativnosti**	**Strokovnost**	**Formalnost**	**Kompleksnost**	**Ponovljivost**	**Strategija realizacije**
Regresijsko testiranje	nizka	nizka	nizka	nizka	visoka	inkrementalna
Sprejemno testiranje	normalna	normalna	visoka	normalna	visoka	inkrementalna, vzporedna, prekrivajoča
Kontinuirano testiranje	visoka	normalna	visoka	normalna	visoka	inkrementalna, vzporedna, prekrivajoča
Destruktivno testiranje	normalna	normalna	nizka	nizka	normalna	inkrementalna, vzporedna, prekrivajoča

Tabela 4.8: Vrednosti projektnih karakteristik alternativ elementa razvojno testiranje

Kriterij	**Regresijsko testiranje**	**Sprejemno testiranje**	**Kontinuirano testiranje**	**Destruktivno testiranje**
Stopnja inovativnosti	1	2	3	2
Strokovnost	3	2	2	2
Formalnost	1	3	3	1
Kompleksnost	3	2	2	3
Ponovljivost	3	3	3	2
Strategija realizacije	0.33	1	1	1
Vsota	**11.33**	**13**	**14**	**11**

Tabela 4.9: Ocene alternativ elementa razvojno testiranje po projektnih karakteristikah

Slika 4.11: Sociološki vidik



Slika 4.12: Tehnični vidik

Analiza zahtev

Element analiza zahtev je tako s sociološkega kot tehničnega vidika ocenjen podpovprečno, z ekonomskega vidika pa, dobro. Na sliki 4.11 vidimo porazdelitev ocen za vprašanje s sociološkega vidika, na sliki 4.12 pa porazdelitev za vprašanje iz tehničnega vidika. Za sociološki vidik je razpon ocen zelo velik. Posledica tega je to, da element uporabniki različno izvajajo in so zato z njim različno zadovoljni. To nameravamo rešiti s predlaganimi alternativami, ki bodo element bolje definirale. Porazdelitev ocen elementa je enakomerna od ocene dva do šest. Ocene za tehnični vidik so bolj enotne. Prevladujejo povprečne ocene, razen enega uporabnika, ki je element ocenil kot tehnično zelo slab. Razlog za nižje ocene je, da trenuten pristop ne omogoča dovolj dobrega analiziranja zahtev. Ker element tehnično ni dobro izveden, posledično tudi sociološko uporabniki z njim niso najbolj zadovoljni.

Iz članka [31] smo identificirali tri alternativne pristope izvajanja analize zahtev. To so analiza zahtev z uporabo naravnega jezika, analiza zahtev z diagrami primerov uporabe in globalna analiza. Alternative so predstavljene v tabeli 4.10.

Ime alternative	**Opis alternative**	**Vključitev v proces podjetja**
Analiza zahtev z uporabo naravnega jezika	Zahteve so po predpisani obliki opisane z naravnim jezikom. Za lažjo kategorizacijo so zahteve razvrščene po skupinah. Tehnika je preprosta in zato lahka za razumevanje številnih deležnikov.	Pri analizi zahtev bi pripravili dokument, v katerem bi zahteve z naravnim jezikom opisali po predpisani obliki in zahteve razvrstili v skupine.
Analiza zahtev z diagrami primerov uporabe	Sestavni deli analize primera uporabe so akterji, primeri uporabe in scenariji primerov uporabe. Rezultat aktivnosti je model, ki vključuje množico diagramov, opisov akterjev, primerov uporabe, scenarijev primerov uporabe in druge potrebne informacije, potrebne za predstavitev primerov uporabe. Primeri uporabe opisujejo splošno obnašanje in ne vključujejo opisov specifičnih akcij.	Vključitev bi bila možna z uporabno diagramov primerov uporabe UML (angl. Unified Modeling Language) pri aktivnosti analiza zahtev.
Globalna analiza	Namen globalne analize je identificirati in analizirati faktorje, ki imajo globalen vpliv na arhitekturo. Faktorji so treh kategorij: tehnološki, organizacijski in projektni. Primer tehnoloških faktorjev so ciljna strojna oprema, platforma programske opreme, standardi in razvojna ogrodja, ki bodo uporabljena. Organizacijski faktorji vključujejo plan in proračun uporabljenih razvojnih procesov ter usposobljenost razvijalcev. Produktni faktorji pa so zahteve, specifične za produkt.	Analizo zahtev bi lahko izvajali z uporabo tabel faktorjev. Za vsako kategorijo faktorjev bi bila svoja tabela, ki bi vsebovala opis faktorja, aspekt faktorja, ki se lahko spremeni, in opis vpliva faktorja ali spremembe faktorja na arhitekturo.

Tabela 4.10: Opis alternativ analize zahtev

Izbrane projektne karakteristike, po katerih bomo ocenjevali alternative, so iz organizacijske dimenzije stopnja inovativnosti, iz človeške dimenzije strokovnost, jasnost in stabilnost ter iz aplikacijske domene formalnost, kompleksnost in spremenljivost. Za uteževanje karakteristik se nismo odločili, ker ocenjujemo, da so vse karakteristike enako pomembne.

Glede na opis projekta in evalvacijo so bile identificirane vrednosti projektnih karakteristik. To so:

- stopnja inovativnosti: normalna,
- strokovnost: normalna,
- formalnost: normalna,
- kompleksnost: nizka,
- spremenljivost: visoka,
- jasnost in stabilnost: visoka.

Namen projekta je razvoj informacijskega sistema za upravljanje skladišča. Projekt je glede na dosedanje projekte ocenjen z normalno stopnjo inovativnosti. Ne prinaša večjih poslovnih ali tehničnih inovacij. Strokovna usposobljenost kadra je dobra, saj jih je veliko že delalo na podobnih projektih. Formalnost standardov za poslovne procese in podporne informacije je normalna. Kompleksnost funkcionalnih komponent sistema je nizka, vendar pa je možna visoka spremenljivost projekta. Cilji, potrebe in želje uporabnikov pri funkcionalnih zahtevah pa so zelo jasne in stabilne.

Stopnje projektnih karakteristik za alternative so prikazane v tabeli 4.11 in izhajajo iz opisov iz članka [31]. Najmanj zahtevna je analiza zahtev z uporabo naravnega jezika. Ta prinaša nizko stopnjo inovativnosti, zahteva nizko strokovno usposobljenost uporabnikov in ni zelo kompleksna. Ker pa so opisi zahtev v naravnem jeziku, to pomeni, da se zahteve lahko hitro spremenijo in prinašajo nizko stopnjo jasnosti in stabilnosti. Alternativa analiza zahtev z diagrami primerov uporabe je malo bolj napredna od tiste z uporabo naravnega jezika. Zato prestavlja večjo stopnjo inovativnosti, je kompleksnejša in bolj formalna. Zahteva višjo stopnjo strokovnosti uporabnikov. Spoznati se morajo z diagrami primerov uporabe. Ta alternativa omogoča visoko stopnjo spremenljivosti in omogoča dobro jasnost in stabilnost funkcionalnih zahtev. Alternativa analize zahtev z globalno analizo izmed vseh predstavljenih alternativ prinaša najvišjo stopnjo inovativnosti, zahteva dobro strokovno usposobljenost uporabnikov, omogoča normalno spremenljivost in formalnost ter omogoča jasnost in stabilnost funkcionalnih zahtev. Ocene so bile dodeljene po postopku, ki je opisan v poglavju 3.3.2. Po seštetih ocenah, ki so prikazane v tabeli 4.12, je bila kot najboljša alternativa izbrana analiza zahtev z uporabo diagramov primerov uporabe.

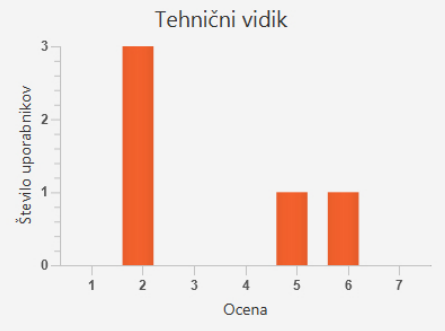
Alternativa	**Stopnja inovativnosti**	**Strokovnost**	**Formalnost**	**Kompleksnost**	**Spremenljivost**	**Jasnost in stabilnost**
Z uporabo naravnega jezika	nizka	nizka	nizka	nizka	visoka	nizka
Diagrami primerov uporabe	normalna	normalna	normalna	normalna	visoka	normalna
Globalna analiza	visoka	visoka	normalna	normalna	normalna	normalna

Tabela 4.11: Vrednosti projektnih karakteristik alternativ elementa analiza zahtev

Kriterij	**Z uporabo naravnega jezika**	**Diagrami primerov uporabe**	**Globalna analiza**
Stopnja inovativnosti	1	2	1
Strokovnost	1	2	2
Formalnost	1	1	2
Kompleksnost	3	2	2
Spremenljivost	3	3	2
Jasnost in stabilnost	1	2	2
Vsota	**10**	**12**	**11**

Tabela 4.12: Ocene alternativ elementa analiza zahtev po projektnih karakteristikah

Slika 4.13: Sociološki vidik



Slika 4.14: Tehnični vidik

Dodeljevanje nalog razvijalcem

Element dodeljevanje nalog razvijalcem je bil tako s sociološkega kot tehničnega vidika ocenjen precej podpovprečno. Z ekonomskega vidika, pa je bil dobro ocenjen. Na sliki 4.13 je predstavljena porazdelitev ocen za vprašanje s sociološkega vidika, na sliki 4.14 pa porazdelitev za vprašanje iz tehničnega vidika. Iz ocen za sociološki vidik opazimo, da sta dva uporabnika zelo zadovoljna z elementom, nekateri ostali so ga ocenili zelo slabo. Pri tehničnem vidiku, pa so kar trije uporabniki element ocenili kot tehnično neučinkovit. Sistem, ki ga v podjetju trenutno uporabljajo za dodeljevanje naloga razvijalcem, je produkt podjetja Atlassian, ki se imenuje JIRA. Z ekonomskega vidika vodstvo ocenjuje, da je orodje učinkovito, uporabniki pa s tehničnega in sociološkega vidika z orodjem niso dovolj zadovoljni. Razlog za slabo oceno bi lahko bil slab pregled nad povezavami med nalogami ali prevelika kompleksnost uporabe orodja.

Kot alternative trenutnemu orodju za dodeljevanje nalog smo identificirali štiri alternativna orodja. To so Trello, Bitrix24, Asana in Freedcamp. Vsa omogočajo dodeljevanje nalog razvijalcem in projektno vodenje projektov. Razlikujejo pa se v kompleksnosti in načinu upravljanja opravil. Predstavitev alternativ je v tabeli 4.13.

Ime alternative	**Opis alternative**	**Vključitev v proces podjetja**
Trello	Trello je spletna aplikacija za projektno vodenje. Orodje za upravljanje projektov uporablja metodologijo Kanban. Projekti so predstavljeni kot table, ki vsebujejo sezname. Seznami vsebujejo kartice, ki predstavljajo posamezna opravila. Kartice potujejo od enega do drugega seznama.	Z vključitvijo orodja Trello v razvojni proces podjetja bi podjetje moralo razvojni proces prilagoditi za izvajanje po metodologiji Kanban. Upravljanje nalog bi potekalo preko kartic z opravili v orodju Trello.
Bitrix24	To je spletna aplikacija, ki omogoča projektno vodenje, upravljanje odnosov s strankami in socialno omrežje. Omogoča kreiranje opravil za uporabnike, določanje odvisnosti med opravili in spremljanje poteka opravil. Omogoča tudi komunikacijo med uporabniki in upravljane z dokumenti.	Z uporabo aplikacije Bitrix24 bi v razvojni proces podjetja vključili orodje, ki ne omogoča samo upravljanja opravil, ampak tudi komunikacijo med uporabniki in upravljanje z dokumenti.
Asana	Asana je spletna aplikacija, ki izboljšuje sodelovanje med ekipami. Vsaka ekipa lahko ustvari delovni prostor, ki vsebuje projekte z opravili. Uporabniki lahko sledijo projektom in opravilom in tako dobijo obvestila ob spremembah.	Z vključitvijo orodja Asana bi vsaka ekipa dobila svoj delovni prostor v orodju. Preko tega, bi lahko kreirala projekte in opravila ter sledila statusu projekta in opravil.
Freedcamp	Freedcamp je brezplačna spletna aplikacija za projektno upravljanje in orodje za sodelovanje med uporabniki. Možno je dokupiti plačljive aplikacije znotraj orodja. Orodje omogoča dodeljevanje nalog uporabnikom, postavljanje mejnikov, tabele za diskusijo in pregled porabljenega časa na izdelavi opravil.	Z uporabo orodja Freedcamp bi podjetje pridobilo brezplačno aplikacijo za dodeljevanje opravil, ki omogoča osnove funkcionalnosti upravljanja opravil.

Tabela 4.13: Opis alternativ aktivnosti dodeljevanje nalog uporabnikom

Projektne karakteristike, izbrane za ocenjevanje alternativ, so iz človeške dimenzije strokovnost, jasnost in stabilnost, iz aplikacijske domene, formalnost in kompleksnost ter iz razvojne strategije sledljivost projekta.

Glede na opis projekta in evalvacijo so bile identificirane želene vrednosti projektnih karakteristik. To so:

- strokovnost: nizka,
- jasnost in stabilnost: visoka,
- formalnost: nizka,
- kompleksnost: nizka,
- sledljivost projekta: dobra.

Namen projekta je razvoj informacijskega sistema za upravljanje skladišč. Projekt zahteva nizko strokovno usposobljenost kadrov. Jasnost in stabilnost ciljev, potreb in želja uporabnikov pri funkcionalnih zahtevah je visoka. Za projekt je potrebna nizka stopnja formalnosti pravil za poslovne procese in podporne informacije. Tudi kompleksnost funkcionalnih zahtev je nizka. Zaradi nizke kompleksnosti je sledljivost projekta dobra.

Stopnje projektnih karakteristik za alternative so bile določene po pregledu opisov alternativ in so prikazane v tabeli 4.14. Najvišja strokovna usposobljenost kadra je potrebna za orodje Bitrix24, ker je to orodje izmed vseh najbolj obsežno. Ostala orodja ne potrebujejo visoke strokovne usposobljenosti kadra. Najboljšo pomoč pri jasnosti in stabilnosti funkcionalnih zahtev nudi orodje Trello, nekoliko nižjo Bitrix24 in Freedcamp in najnižjo orodje Asana. Pri zagotavljanju nižje stopnje formalnosti pravil poslovnih procesov in nižji kompleksnosti sistema nudita največ podpore orodji Trello in Asana, nekoliko manj orodje Freedcamp in najmanj orodje Bitrix24. Za sledljivost projekta so se dobro izkazala orodja Trello, Bitrix24 in Freedcamp. Nekoliko slabše se je izkazalo orodje Asana, ki ne ponuja najboljšega pregleda nad potekom projekta.

Ocene so bile dodeljene po postopku, ki je opisan v poglavju 3.3.2. Po seštetih ocenah, ki so prikazane v tabeli 4.15, je bila kot najboljša alternativa izbrano orodje Trello. V tem primeru smo karakteristike še utežili, ker smo želeli analizirati relativno pomembnost kriterijev. Uteži je določilo vodstvo podjetja z analizo pomembnosti. Pri analizi pomembnosti so razvrstili kriterije glede na njihovo pomembnost. Nato so vsakemu kriteriju dodelili neko vrednost od 1 do 100. Na koncu smo iz teh vrednosti preračunali relativne pomembnosti kriterijev tako, da smo jih pretvorili v vrednosti od 0 do 1.

Alternativa	**Strokovnost**	**Jasnost in stabilnost**	**Formalnost**	**Kompleksnost**	**Sledljivost projekta**
Trello	nizka	visoka	nizka	nizka	dobra
Bitrix24	visoka	normalna	visoka	visoka	dobra
Asana	nizka	nizka	nizka	nizka	slaba
Freedcamp	nizka	normalna	normalna	normalna	dobra

Tabela 4.14: Vrednosti projektnih karakteristik alternativ elementa dodeljevanje nalog razvijalcem

Kriterij	**Utež**	**Trello**	**Bitrix24**	**Asana**	**Freedcamp**
Strokovnost	0,1	3	1	3	3
Jasnost in stabilnost	0,3	3	2	1	2
Formalnost	0,1	3	1	3	2
Kompleksnost	0,2	3	1	3	2
Sledljivost projekta	0,3	1	1	0	1
Vsota		**2,4**	**1,3**	**1,5**	**1,8**

Tabela 4.15: Ocene alternativ elementa dodeljevanje nalog razvijalcem po projektnih karakteristikah

4.4.3 Rezultati potrditvenega dela

Vodstvu podjetja smo predstavili rezultate, ki smo jih pridobili z uporabo našega ogrodja. Pri aktivnosti razvojno testiranje, smo jim predlagali razširitev testiranja z uporabo kontinuiranega testiranja. Pri tem testiranju se kot del razvojnega procesa izvajajo tudi avtomatski testi. Vodstvu podjetja se zdi vpeljava avtomatskih testov dober predlog. Strinjajo se, da aktivnost razvojnega testiranja ne poteka dovolj dobro in bi jo bilo treba izboljšati. Nekaj avtomatskih testov uporabljajo že sedaj. V prihodnosti nameravajo dodati še večje število avtomatskih testov. Pri analizi zahtev smo vodstvu predlagali, da izvajajo analizo zahtev z uporabo diagramov primerov uporabe. Vodstvo podjetja se strinja, da bi uporaba diagramov primerov uporabe pripomogla k večji jasnosti zahtev. Pri prihodnjih projektih nameravajo za izvajanje aktivnosti analiza zahtev, uporabiti tudi diagrame primerov uporabe. Zadnji predlog, ki smo ga podali podjetju, je, da za dodeljevanje nalog razvijalcem uporabijo orodje Trello. Vodstvo podjetja se je odločilo, da bo še vedno uporabljalo orodje JIRA, za dodeljevanje nalog razvijalcem. Bodo pa v orodje vključili modul za uporabo agilnih metodologij. Po zgledu metodologije Kanban bodo uporabili tablo za upravljanje z opravili. Poleg tega bodo uporabili tudi nekaj elementov metodologije Scrum.

Vodstvo podjetja je potrdilo, da je z uporabo našega ogrodja, pridobilo netrivialne informacije, ki jim bodo pomagale pri izboljšavi procesa razvoja informacijskih sistemov. Dva od treh predlogov, ki smo jih predlagali, bodo v prihodnosti uporabili v svojem razvojnem procesu. Za predlog o zamenjavi orodja za dodeljevanje nalog razvijalcem se niso odločili. Bodo pa orodje, ki ga trenutno uporabljajo, razširili s funkcionalnostmi, ki so vključene v orodju, ki smo jim ga predlagali.

Poglavje 5

Sklepne ugotovitve

Glavni namen magistrske naloge je bil razviti ogrodje, ki omogoča izbiro najbolj primernih pristopov za izboljšavo procesa razvoja informacijskih sistemov. Razvito ogrodje združuje področji evalvacijskih modelov in konstruiranja metodologij ter omogoča celovito evalvacijo trenutnega procesa razvoja informacijskih sistemov v podjetju in izbor najbolj primernih pristopov za izboljšavo. Pomemben doprinos ogrodja je, da zna rezultate evalvacijskega dela uporabiti kot vhod v izbirni del ogrodja. Izbirni del uporablja tehniko s področja konstruiranja metodologij in omogoča izbiro najbolj primernih pristopov za izboljšavo procesa razvoja informacijskih sistemov.

S študijo primera smo preverili delovanje ogrodja na primeru podjetja. V okviru študije smo najprej pregledali trenutni potek izvajanja procesa razvoja informacijskih sistemov in identificirali elemente razvojnega procesa. Nato smo evalvirali elemente razvojnega procesa, kasneje pa za nekatere slabše ocenjene elemente pripravili predloge izboljšav. Podjetju smo predstavili tri predloge izboljšav, ki so bili pridobljeni z našim ogrodjem. Podjetje ugotavlja, da so z uporabo našega ogrodja pridobili netrivialne informacije, ki jim bodo pomagale izboljšati razvojni proces. Dva predloga bo podjetje v celoti vključilo v razvojni proces, enega pa delno.

Ogrodje ima nekaj omejitev. V ogrodju uporabljamo repozitorij elementov procesa razvoja informacijskih sistemov, označenih z metapodatki. V okviru magistrske naloge smo repozitorij realizirali le v omejenem obsegu, tj. le za elemente, ki smo jih analizirali v okviru študije primera. Ogrodje je splošno uporabno, vendar se v tej fazi pričakuje, da repozitorij pripravi uporabnik sproti. Možna izboljšava ogrodja bi bila priprava večjega repozitorija, ki bi vseboval več ocenjenih alternativnih elementov, ki izhajajo iz različnih obstoječih metodologij. V pripravljalnem delu ogrodja identificiramo elemente procesa razvoja informacijskih sistemov in jih kasneje evalviramo v evalvacijskem delu. V izbirnem delu ogrodja za te evalvirane elemente potrebujemo alternativne elemente, ki omogočajo

izbiro najprimernejših alternativ za evalvirane elemente. Večji repozitorij alternativnih elementov, bi omogočal izbiro alternativ za več različnih evalviranih elementov. Študija primera je bila izvedena na enem podjetju. V prihodnosti bi bilo smiselno ogrodje preveriti še na več drugih podjetjih.

Literatura

[1] D. Vavpotič, M. Bajec, An approach for concurrent evaluation of technical and social aspects of software development methodologies, Information and Software Technology 51 (2009) 528 – 545.

[2] D. Vavpotič, T. Hovelja, Improving the evaluation of software development methodology adoption and its impact on enterprise performance, Computer Science and Information Systems 9 (1) (2012) 165 – 187.

[3] C. P. Team, Cmmi for development, version 1.3, in: Tehnično poročilo CMU/SEI-2010-TR-033, Software Engineering Institute, Carnegie Mellon University, 2010.

[4] C. K. Riemenschneider, B. Hardgrave, F. D. Davis, Explaining software acceptance of methodologies: a comparison of five theoretical models, IEEE Transactions on Software Engineering 28 (2002) 1135 – 1145.

[5] M. Niazi, D. Wilson, D. Zowghi, A maturity model for the implementation of software process improvement: an empirical study, Journal of Systems and Software 74 (2005) 155 – 172.

[6] S. Sharma, A. Rai, An assessment of the relationship between isd leadership characteristics and is innovation adoption in organizations, Information and Management 40 (2003) 391 – 401.

[7] B. Fitzgerald, Formalized system development methodologies: a critical perspective, Information Systems Journal (1996) 3–23.

[8] W. H. DeLone, R. McLean, The delone and mclean model of information systems success: A ten-year update, Journal of Management Information Systems 19 (4) (2003) 9 – 30.

[9] B. Henderson-Sellers, J. Ralyte, P. J. Agerfalk, M. Rossi, in: Situational Method Engineering, 2014.

[10] S. Brinkkemper, Personal email communication to authors, 2006.

[11] OMG, Software and system process engineering metamodel (v2.0), 2007.

[12] I. Ruiz-Rube, J. M. Dodero, M. Palomo-Duarte, Uses and applications of spem process models. a systematic mapping study, Journal of Software Maintenance and Evolution: Research and Practise 1 (32) (2012) 999 – 1025.

[13] H. Agh, R. Ramsin, A pattern-based model-driven approach for situational method engineering, Information and Software Technology 78 (2016) 95 – 120.

[14] A. F. Harmsen, S. Brinkkemper, H. Oei, Situational method engineering for information systems projects. in methods and associated tools for the information systems life cycle, Proceedings of the IFIP WG8.1 Working Conference (1994) 169 – 194.

[15] J. J. Odell, Introduction to method engineering, Vol. 5, 1995.

[16] D. Firesmith, B. Henderson-Sellers, I. Graham, in: OPEN Modeling Language, 1997, p. 276.

[17] B. Henderson-Sellers, J. Ralyte, Situational method engineering: State-of-the-art review, Journal of Universal Computer Science 16 (3) (2010) 424 – 478.

[18] J. Ralyte, R. Deneckere, C. Rolland, Towards a generic method for situational method engineering, in: Advanced Information Systems Engineering, 2003, pp. 95–110.

[19] J. Ralyte, C. Rolland, R. Deneckere, Towards a meta-tool for change-centric method engineering: A typology of generic operators, in: Proceeding of CAiSE, 2004, pp. 202–218.

[20] E. Kornyshova, R. Deneckere, C. Salinesi, Method chunks selection by multicriteria techniques: an extension of the assembly-based approach, IFIP International Federation for Information Processing 244 (2007) 64 – 710.

[21] K. van Slooten, K. Hodes, Characterizing is development project, in: Proceedings of IFIP TC8 Working Conf. on Method Engineering: Principle of Method Construction and Tool Support, 1996, pp. 29–44.

[22] B. Henderson-Sellers, J. M. Edwards, in: BOOKTWO of Object-Oriented Knowledge: The Working Object, 1994, p. 594.

[23] I. Graham, in: Migrating to Object Techology, 1995, p. 552.

[24] V. P. Nguyen, B. Henderson-Sellers, Towards automated support for method engineering with the open approach, in: Proceddings of the 7th IASTED Sea Conference, 2003, pp. 691–696.

[25] V. Seidita, J. Ralyte, B. Henderson-Sellers, M. Cossertino, N. Arni-Bloch, A comparison of deontic matrices, maps and activity diagrams for the construction of situational methods, 2007.

[26] S. Brinkkemper, M. Saeki, F.Harmsen, Meta-modelling based assembly techniques for situational method engineering, Information Systems 24 (3) (1999) 209 – 228.

[27] I. Mirbel, J. Ralyte, Situational method engineering: Combining assembly-based and roadmap-driven approaches, Requirements Engineering 1 (2006) 58 – 78.

[28] J. Mustajoki, R. P. Hamalainen, A. Salo, Decison support by interval smart/swing - incorporating imprecision in the smart and swing methods, Decision Sciences 36 (2) (2005) 317 – 339.

[29] M. Poyhonen, R. P. Hamalainen, On the convergence of multiattribute weighting methods, European Journal of Operational Research 129 (3) (2001) 569 – 585.

[30] R. L. Keeney, Foundations for making smart decisions, IIE Solutions 31 (5) (1999) 24 – 30.

[31] L. Bass, J. Bergery, P. Clements, P. Merson, I. Ozkaya, R. Sangwan, A comparison of requirements specification methods from a software architecute perspective, 2006.